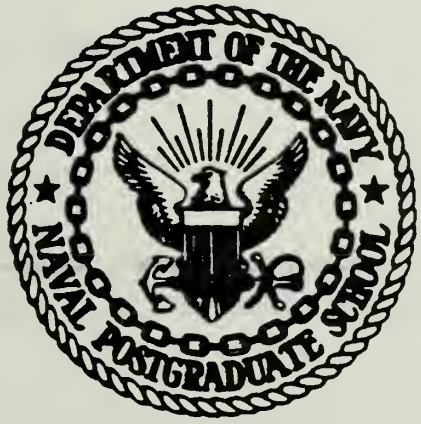
COMPUTATIONAL ADVANCES IN LARGE-SCALE NON-LINEAR OPTIMIZATION.

Dennis Ross Dean

NAVAL POSTGRADUATE SCHOOL
Monterey, California



THESIS

COMPUTATIONAL ADVANCES IN LARGE-SCALE

NONLINEAR OPTIMIZATION

by

Dennis Ross Dean

September 1981

Thesis Advisor: G. G. Brown

Approved for public release, distribution unlimited

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
Computational Advances in Large-Scale Nonlinear Optimization

5. TYPE OF REPORT & PERIOD COVERED
Master's Thesis
September 1981

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)
Dennis Ross Dean

8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Naval Postgraduate School
Monterey, California 93940

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS
Naval Postgraduate School
Monterey, California 93940

12. REPORT DATE
September 1981

13. NUMBER OF PAGES
134

14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)

15. SECURITY CLASS. (of this report)
Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)
Approved for public release, distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)
Large-scale nonlinear programming, optimization, large-scale linear programming, mixed integer programming, algorithm evaluation, nonlinear test problems.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)
This is a comparison of two state-of-the-art large-scale nonlinear optimization systems exhibiting unprecedented problem solution capabilities both in size of problem handled and method of solution. These codes are MINOS, developed by B. A. Murtagh and M. A. Saunders, and XS, developed by G. G. Brown and G. W. Graves. The codes are evaluated with respect to their problem solving capabilities and potential for practical application by analysts. Computational results are presented for thirteen

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

nonlinear and nonlinear mixed integer test problems with from two to 793 variables (12 to 100 integer variables) and one to 401 constraints. Portions of this work were presented at the CORS/ORSA/TIMS joint meeting in Toronto, May 1981.

Approved for public release, distribution unlimited.

Computational Advances in Large-Scale Nonlinear Optimization

by

Dennis Ross Dean
Lieutenant Commander, United States Navy
B.S., Purdue University, 1972

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
September 1981

ABSTRACT

This is a comparison of two state-of-the-art large-scale nonlinear optimization systems exhibiting unprecedented problem solution capabilities both in size of problem handled and method of solution. These codes are MINOS, developed by B. A. Murtagh and M. A. Saunders, and XS, developed by G. G. Brown and G. W. Graves. The codes are evaluated with respect to their problem solving capabilities and potential for practical application by analysts. Computational results are presented for thirteen nonlinear and nonlinear mixed integer test problems with from two to 793 variables (12 to 100 integer variables) and one to 401 constraints. Portions of this work were presented at the CORS/ORSA/TIMS joint meeting in Toronto, May 1981.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENT

I would like to thank Mike Saunders who so graciously provided us with an "early" version of his code even before all the documentation was ready and assisted us in getting MINOS running. My thanks also go to Glenn Graves, who very patiently put up with the use and abuse of his godchild, the X System. And finally, to Jerry Brown, whose patience, teaching skill, humor, and insights made this all worthwhile and whose friendship is even more valuable . . .

THANK YOU, Jerry!

I. INTRODUCTION

This study is a comparison of two state-of-the-art nonlinear programming codes that are designed to accommodate problems of thousands of variables and constraints. While there are many codes designed to handle the general linear programming (LP) problem and its specializations, there are significantly fewer systems designed to reliably solve the much more difficult nonlinear programming (NLP) problem. Of these, very few are capable of solving "large-scale" problems: larger than, say, a thousand constraints or a thousand variables or more. Most of these large-scale codes are internal, proprietary systems developed by companies for the solution of their specific industrially related problems; the codes used by petroleum refiners for chemical process control provide singular examples of such contributions.

One of the codes evaluated in this study is MINOS (Modular In-core Nonlinear Optimization System) developed by B. A. Murtagh, the University of New South Wales, and M. A. Saunders, Stanford University. The other is XS (X System) developed by G. G. Brown, Naval Postgraduate School, and G. W. Graves, University of California at Los Angeles.

This is the first independent comparison of either code and is intended to serve both as an evaluation of each and as a guide to the potential user concerned with the applicability of each code to the individual problem with which he might be faced.

Two caveats should be kept in mind while reading this evaluation. First, the codes are quite different in intended use. MINOS is intended

as an academic production code and is designed to be readily distributed and applied by a wide variety of users. Extensive documentation and reliable performance have been paramount concerns in the development of MINOS. On the other hand, XS is used as an advanced experimental testbed for optimization research. The fully instrumented version used in this comparison is a prototype designed to be used almost exclusively by its originators and their co-workers for a wide range of problems, such as large mixed integer and linear formulations and especially for decomposition problems. As such, it is in a continual state of flux and varies considerably in its content (hopefully in an improving direction) from month to month. All results from XS are from the most recent prototype system at the time of publication cutoff for this thesis with no specialization for nonlinear programming. Academic and industrial production versions of XS are typically customized to the application at hand and thoroughly documented for routine use.

Second, although both systems are "large-scale" nonlinear codes which have been successfully used on many large, real-life problems, because of their intended day-to-day application, their characteristics are not the same, nor are they intended to be. Therefore, any differences between them in speed or capability may be attributable to design intention rather than relative deficiencies in the algorithms, underlying data structures, or implementation.

A. GENERAL PROBLEM STATEMENT

The general linear programming (LP) problem can be stated as:

minimize $c^T x$ (objective function)

subject to $\underline{r} \leq A\,x \leq \bar{r}$ (ranged constraints)

$\underline{b} \leq x \leq \bar{b}$ (bounds on variables)

where:

x = variables;

c^T = cost coefficients;

A = constraint matrix coefficients;

$\underline{r}$, $\bar{r}$ = upper and lower constraint ranges;

$\underline{b}$, $\bar{b}$ = upper and lower variable bounds.

The general non-separable nonlinear (NLP) problem can be stated as:

minimize $f(x)$ (objective function)

subject to $\underline{r} \leq g(x) \leq \bar{r}$ (ranged constraints)

$\underline{b} \leq x \leq \bar{b}$ (bounds on variables)

where

x = variables;

$f(x)$ = general non-separable, nonlinear function;

$g(x)$ = general non-separable, nonlinear constraint;

$\underline{r}$, $\bar{r}$ = upper and lower constraint ranges;

$\underline{b}$, $\bar{b}$ = upper and lower variable bounds.

B. COMPARISON CRITERIA

In any study of this nature, one of the primary concerns is the criteria with which the codes are to be objectively compared. In this case, the guidelines recently published in Operations Research [Ref. 1] will be used with some modification to prevent comparisons that are not valid because of the somewhat different nature of the two codes. These criteria, to be elaborated in Chapter IV, are listed below.

1. Algorithm Capabilities

This section contains a general overview of the types and classes of problems for which each code is designed and comments on the growth capabilities of each code.

2. CPU (Compute) Time

The CPU times listed are the virtual CPU times required for each problem running with precompiled load modules for each primary system code and do not include the linkage editor times. Since the problems have been run interactively on a virtual memory computer system, these times will vary somewhat from run to run depending upon computer loading. Extensive experience on the host computer with these problems indicates that the listed CPU times are valid within one per cent. Although the actual "clock," or "response," time (as opposed to CPU time) varies as a function of system loading; empirical evidence gathered while conducting this study suggests that a useful rule-of-thumb is that actual clock time is approximately four times as long as CPU time for the virtual memory time-sharing system used. This should provide a reasonable estimate of the response times to be expected for problems of this study.

Because of the region requirements of the FORTRAN compiler used on the host computer (see Section I.C), one megabyte of default virtual memory was used for all problems in a single-step procedure. Comments concerning problem-dependent memory requirements of each system will be made in Chapter IV.

3. Number of Iterations

The number of major iterations (linearizations) and pivots required to reach solution is given for each problem, with the caveat

that the nature of an "iteration" varies considerably between the algorithms. The specific nature of these iterations is discussed in Sections II.A.1 and II.B.1.

4. Number of Function Evaluations

The number of function evaluations to reach solution is listed for each algorithm. However, since this number includes both objective function and constraint evaluations, as well as gradient calculations in the case of MINOS, different amounts of information may be obtained on each function call and this may, therefore, be a deceptive comparison.

5. User Friendliness

One of the primary goals of this study is to evaluate the ability of a user familiar with some optimization theory but with little experience with the individual codes to set up and successfully solve a problem. Because of the codes' different design motivations, it was expected from the beginning of the study that MINOS would be far superior in this regard.

a. Ease of Setup

One measure of the flexibility of a problem-solving system is the ease with which it can be adapted by the general user to the particular problem/data structure at hand.

b. Debug Output

During initial debugging of a problem, varying quantities and types of diagnostic information may be required to isolate a particular error. The ability of each code to provide a tailored output in concise, readable form for the user will be evaluated.

c. Failure Mode

Since the perfect optimization code has yet to be developed, one measure of a code's performance is its ability to fail "gracefully," leaving the user in a posture from which he can recover without all of his effort being wasted. The information given to the user when each code fails is examined to evaluate its usefulness in further problem exploration.

d. Robustness

As an aid to the inexperienced user, codes should be robust in their default parameters to minimize the amount of "tuning" that need be done on most problems. At the same time, the parameters must have sufficient scope and power to allow the experienced user to exploit the structure of difficult problems where interaction by the analyst is required in order to achieve a solution.

C. TEST PROBLEMS

This study has been handicapped, as have other similar efforts, by the lack of suitable test problems to test the full capabilities of the codes. In this case, the need has been for large (several hundred variables or more) real-life problems. The author has been unable to secure such problems for which publication release is available and for which the data is in a form that is readily usable by both codes. This is a continuing and widely recognized problem which has prompted the development of a number of nonlinear artificial problem generators. However, although these generators can produce arbitrarily large problems, it has been the experience of the developers of the X System that the randomly generated problems produced by the generators are

not realistic tests for optimization codes because they possess none of the specialized structure that is routinely found in real-life problems and which, in fact, is the very thing capitalized upon by good codes to produce their excellent results on large problems.

Therefore, for the purposes of this test, the emphasis has been placed on real-life, or at least well-known problems, as opposed to generated problems, at the sacrifice of size. Thirteen problems were selected for this study, of which at least nine have been previously published. The variables in the problems range from two to 793 and the constraints from one to 401. The problems contain a mix of linear, nonlinear, equality, and inequality constraints. Also included are two problems with (12 and 100) integer variables which have never before been formally solved as nonlinear integer and nonlinear mixed integer programs.

D. COMPUTER SYSTEM

All computing has been completed in the W. R. Church Computer Center of the Naval Postgraduate School, Monterey, California on its installed IBM 3033 computers using the VM/SP timesharing system. The load modules for both optimization systems and for each respective problem generation subroutine were generated with the FORTRAN IV (H Extended) compiler using the OPTIMIZE (2) option [Ref. 2]. All problems have been solved interactively in real-time on the computer system using precompiled load modules of the respective optimization systems linked with code and parameter data for the individual problems. This research has promoted development of an extensive real-time library of service routines to aid in the preparation, execution, and interpretation of large optimization

problems. Although neither optimization system has a truly interactive solution algorithm, each can be used with interactive parameter settings and with real-time monitoring of solution progress, providing a fertile research environment.

II. DESCRIPTION OF SYSTEMS

A. MINOS

The MINOS version discussed in this study is MINOS/AUGMENTED (alias MINOS Version 4.0) which has been developed as an extension of an earlier MINOS vintage which solved problems with nonlinear objective functions, but with strictly linear constraints. MINOS/AUGMENTED (henceforth called MINOS) is a general-purpose nonlinear programming system designed to solve large-scale optimization problems exhibiting linear and nonlinear constraints, linear and nonlinear variables, and exploiting sparsity (relatively few non-null constraint-variable interactions) and exclusive linearity of some constraints and variables (respectively expressed in no nonlinear terms). Nonlinear functions in a problem should be continuous with continuous first derivatives, but need not be separable. Integer variables are not accommodated. The user specifies nonlinear objective functions with one FORTRAN subroutine, nonlinear constraint functions with a second FORTRAN subroutine, while the linear portions of the objective function and constraints, ranges, bounds, and initial starting point (if any) are specified in standard "MPS Format" [Ref. 3].

MINOS employs an augmented Lagrangian algorithm to solve problems with nonlinear constraints. This algorithm uses a sequence of sparse, linearly constrained subproblems which are solved using a reduced-gradient algorithm.

MINOS is intended (as cautioned by its developers) as an extension of (not a replacement for) commercial mathematical programming systems.

Consequently, MINOS does not possess many of the algorithmic options (e.g., dual simplex) or data revision and file handling capabilities common to various commercial optimization systems. A complete description of the code can be found in the support documentation [Ref. 4, 5, 6].

1. Algorithm

A discussion of the MINOS algorithm is logically divided into two cases:

a. Linear Constraints

Where only linear constraints are present, MINOS is designed to solve problems of the form:

$$\text{minimize} \quad f(x) + c^T x$$

$$\text{subject to} \quad Ax \gtreqless b$$

$$\underline{b} \leq x \leq \bar{b}$$

where $f(x)$ is a continuous, continuously differentiable function with gradient:

$$\nabla f(x) = (\partial f / \partial x_j) = g(x).$$

In general, the constraint matrix A is assumed to be large and sparse.

The foundation of MINOS is an efficient and reliable implementation of the revised simplex method for LP [Ref. 7]. This combines sparse matrix technology [Ref. 8: pp. 213-226] with stable numerical methods for computing and modifying a triangular LU factorization of the basis matrix B. A sparse LU factorization of the basis matrix is computed using the "bump and spike" algorithm of Hellerman and Rarick [Ref. 9: pp. 67-76], which is updated in a stable manner by the method of Bartels and Golub [Ref. 10: pp. 266-268].

In order to extend the simplex method for (LP) to (NLP), superbasic variables are defined in addition to the usual basic (dependent) and nonbasic (independent) variables. Both basic and superbasic variables may possess non-extremal values (between their respective bounds). In the reduced-gradient technique employed [Ref. 11: pp. 97-131], at a given iteration there are NS (Number of Superbasic) superbasic variables. They are free to move in any desirable direction which improves the value of the objective function, while the basic variables are adjusted to maintain feasibility with respect to linear constraints. If no improvement can apparently be made with the current set of superbasics, one (or more) of the nonbasic variables is selected to become superbasic. This increases NS and the process is repeated. If, at any time, a basic or superbasic variable reaches one of its bounds, that variable becomes nonbasic and NS is reduced by 1. By the usual "pricing" of the nonbasic columns, Lagrange multipliers for the current active constraints are obtained, which then indicate which nonbasic variables (if any) should be released from their bounds. If required, they are moved from the nonbasic to the superbasic set, rather than from nonbasic to basic as in the more conventional simplex method.

A stable implementation of a quasi-Newton method is used to optimize over the superbasic variables. This method uses a triangular matrix of dimension NS to approximate the reduced Hessian (a suitably transformed sub-section of the matrix of second derivatives, $(\partial^2 f/\partial x_i \partial x_j)$). In large problems (i.e., NS grows as large as 100 or 200), the data region required by the quasi-Newton method becomes excessive and MINOS automatically substitutes the Fletcher-Reeves

Conjugate Gradient Method [Ref. 12: pp. 149-154], which consumes relatively less memory. The rate of convergence of the algorithm drops significantly, but must be accepted due to storage limitations.

The linear constraint problem may be re-formulated as:

$$\text{minimize} \quad f(x_N) - s_{obj}$$

$$\text{subject to}$$

$$\bar{P}: \quad [A_N \; A_L \; b \; I] \begin{bmatrix} x_N \\ x_L \\ \rho \\ s \end{bmatrix} = 0, \; \underline{b} \leq \begin{bmatrix} x_N \\ x_L \\ \rho \\ s \end{bmatrix} \leq \bar{b}$$

$$\text{given} \quad \nabla f(x_N) = g(x_N)$$

where:

- x_N = the nonlinear variables (those that are directly involved in the function $f(x_N)$);
- x_L = the linear variables (the remaining part of x);
- ρ = the right-hand side (RHS) variable, which has upper and lower bounds of -1.0;
- s = the slack or logical variables (one for each row of A);
- s_{obj} = $c_L x_L^T + c_0$ (linear objective function value);
- A = $[A_N \; A_L]$ linear coefficients, partitioned as x_N and x_L;
- b = right hand side (RHS), composition or $\underline{r}$ and $\bar{r}$, where:

$$b = \begin{cases} \bar{r} & ; \text{ if } A_N x_N + A_L x_L > \bar{r} \\ \underline{r} & ; \text{ if } A_N x_N + A_L x_L < \bar{r} \\ \underline{r} \text{ or } \bar{r} & ; \text{ otherwise;} \end{cases}$$

- m = the number of rows in A;
- n = the number of columns in A;

. nn = the number of nonlinear variables, the number of terms in x_N;

. NS = the number of superbasic variables.

At any particular stage, the n + 1 + m columns of [A b I] are implicitly ordered as shown in Figure 1. The nonlinear variables may end up anywhere in B,S, or N.

b. Nonlinear Constraints

When a problem contains nonlinear constraints, MINOS does not necessarily satisfy the nonlinear constraints until an optimal solution is achieved. Therefore, the nonlinear constraint functions may need to be defined outside their ranges.

The problem must be expressed for MINOS in the following standard form:

$$\text{minimize} \qquad f^0(x) + c^T x + d^T y \text{ (nonlinear objective)} \qquad (1)$$

$$\text{subject to} \qquad f(x) + A_1 y = b_1 \text{ (nonlinear constraints)} \qquad (2)$$

$$A_2 x + A_3 y = b_2 \text{ (linear constraints)} \qquad (3)$$

$$\underline{b} \leq \begin{bmatrix} x \\ y \end{bmatrix} \leq \bar{b} \quad \text{(bounds on variables)} \qquad (4)$$

where: $$f(x) = \begin{bmatrix} f^1(x) \\ \vdots \\ f^m 1(x) \end{bmatrix};$$

and the functions f(x) should be smooth and have computable gradients. The components of x are the nonlinear variables and must preceed the linear variables y in the problem. The constraints (2) are the nonlinear constraints and must appear in the problem before the linear constraints (3). The general constraints may contain any type of inequality, and

	m	NS	n + 1 - NS
[A b I] $\hat{P}$ =	B	S	N
	Basic	Superbasic	Nonbasic

where: $\hat{P}$ is a column permutation transformation of $\bar{P}$.

Fig. 1. MINOS Tableau Arrangement

ranges may be defined for the constraints. Upper and lower bounds (4) may (and should!) be specified for all variables.

The solution process [Ref. 13] consists of a sequence of "major iterations;" at the beginning of each the nonlinear constraints are linearized at a current point x_k, approximated to first-order by:

$$\tilde{f}(x, x_k) = f(x_k) + J(x_k)(x - x_k) ,$$

which can be written as:

$$\tilde{f} = f_k + J_k(x - x_k) . \tag{5}$$

Here, J(x) is the Jacobian matrix whose ij^{th} element is $\partial f^i(x)/\partial x_j$. The objective function is also modified, producing the following subproblem:

$$\text{minimize} \quad f^0(x) + c^T x + d^T y - \lambda_k^T(f - \tilde{f}) + (1/2)\rho(f - \tilde{f})^T(f - \tilde{f}) \quad \text{(quadratic objective function)} \tag{6}$$

$$\text{subject to} \quad \tilde{f} + A_1 y = b_1 \quad \text{(linearized constraints)} \tag{7}$$

$$A_2 x + A_3 y = b_2 \quad \text{(linear constraints)} \tag{8}$$

$$\underline{b} \leq \begin{bmatrix} x \\ y \end{bmatrix} \leq \bar{b} \quad \text{(bounds on variables).} \tag{9}$$

The objective function (6) is called an augmented Lagrangian. The vector λ_k is an estimate of the Lagrange multipliers for the nonlinear constraints, and the term involving ρ is a modified quadratic penalty function. If desired, the Lagrangian term of the modified objective function may be set to zero by the user. The penalty parameter ρ may also be controlled by the user. The problem (6)-(9) is, of course, stated in precisely the form required by the linearly constrained MINOS algorithm.

A flowchart of the MINOS NLP solution process is given in Figure 2.

If (x_k, λ_k) are the final solution and multiplier estimates from the k^{th} subproblem, convergence is assumed to have occurred if the following conditions are true:

- x_k is an optimal solution to the subproblem;
- x_k satisfies the nonlinear constraints within a specified tolerance;
- λ_k is not substantially different from λ_{k-1};
- x_{k+1} is an optimal solution to its subproblem;
- a basis change did not occur during solution of subproblem k + 1;
- the reduced gradient did not increase significantly during solution of that subproblem.

The salient point is that x_k is checked for feasibility and then the final point x_{k+1} is checked for optimality. Since normally very few basis changes occur on the final subproblem (ideally none), the solutions will be virtually identical and the tests for feasibility and optimality will have been applied to essentially the same point.

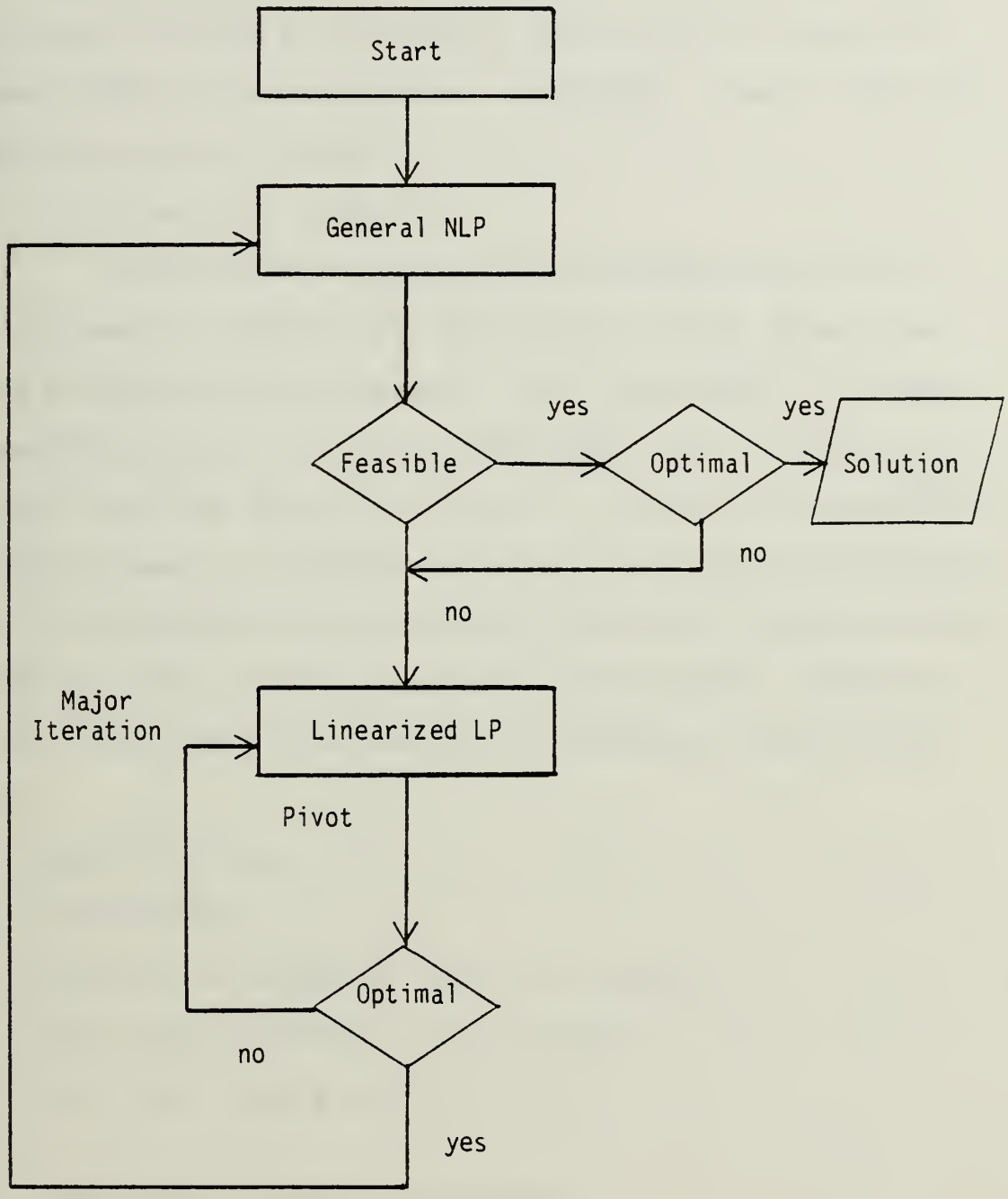


Fig. 2. MINOS Iteration Flowchart

2. Code Structure

MINOS is intended for use primarily as a "stand-alone" system which solves a problem or a sequence of problems and then terminates; however, MINOS can also be called as a subprogram. Figure 3 shows the subroutine structure of MINOS.

a. Stand-Alone Structure

Some problem description and code tuning are provided to MINOS by means of a SPECS file which contains a list of keywords and values to define run-time parameters. Most of the data for a problem is provided by means of the standard MPS Format file. If the problem contains a nonlinear objective function, it is specified (as can be its gradient) by means of a FORTRAN subroutine called CALCFG provided by the user. If the problem contains nonlinear constraints, they are provided by the user (along with their gradients) in the form of a FORTRAN subroutine called CALCON. The input data is processed in the following order:

. the SPECS file;

. the MPS file;

. a basis file (allows an initial basis crash);

. data read by CALCON on its first entry;

. data read by CALCFG on its first entry;

. . .

. data read by CALCFG on its last entry;

. data read by CALCON on its last entry.

The MAIN program provides a single array of storage for the code. The GO subroutine is a control routine which calls subroutine MINOS for each problem to be solved in the input stream. This routine

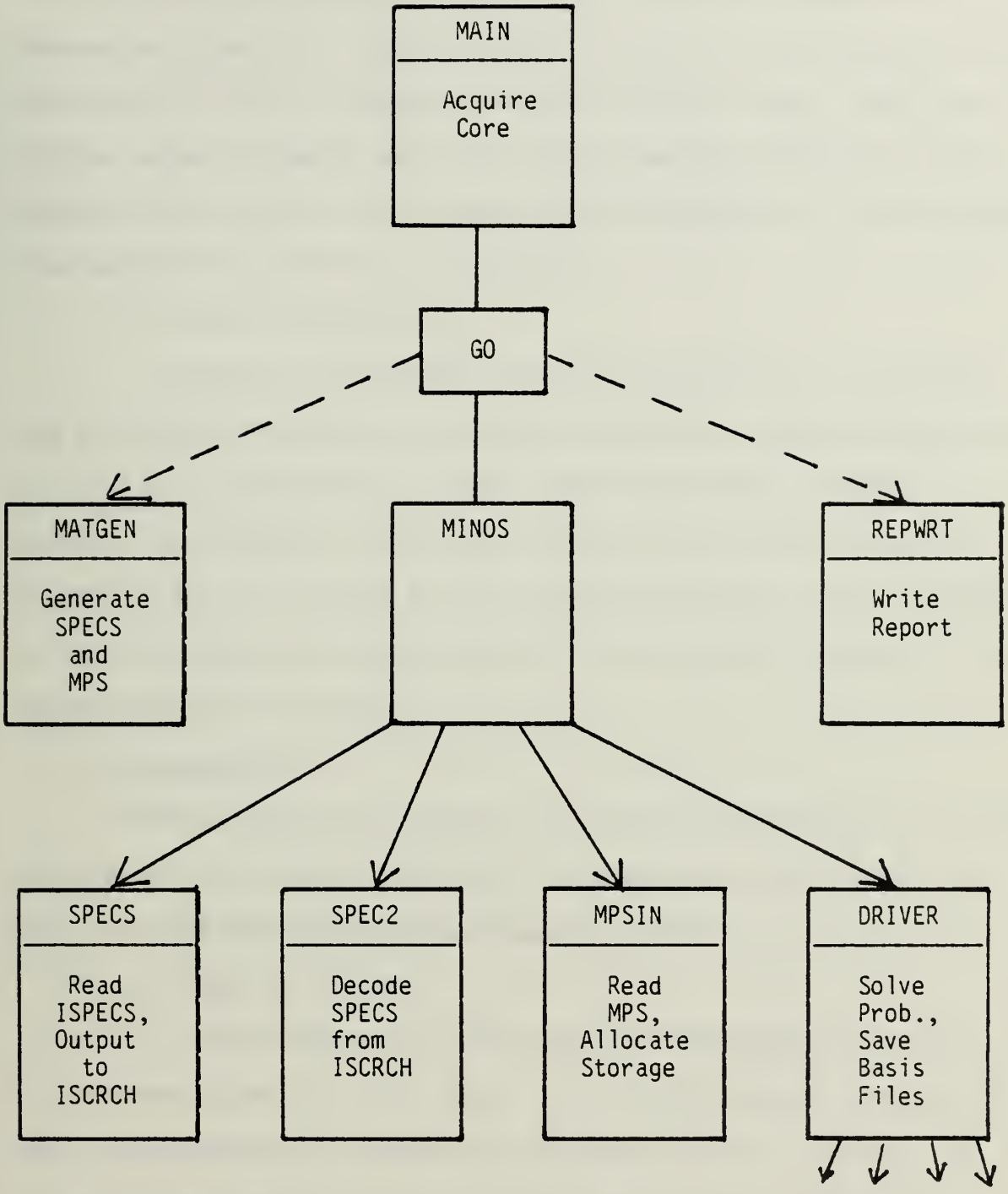


Fig. 3. MINOS Subroutine Structure

can be modified for individual applications. Subroutine MINOS and its sub-programs communicate with the input files supplied to the code by the user (MPS and SPECS) and generate BASIS and SOLUTION files. MINOS also contains output parameters which define the stopping condition for the problem, the dimensions of the problem, and the positions in the workspace array where various subarrays are located.

b. Use as a Subroutine

Because of the modular nature of the code, it is relatively easy to modify the system for individual applications. Most modifications to tailor the system would be to the control routine GO; run-time parameters and constraint information would still be input via the MPS and SPECS files. The dotted lines in Figure 3 show how a Matrix Generator and Report Writer could be incorporated into the system to be used in any manner convenient to the user.

3. Documentation

MINOS documentation consists of three publications [Ref. 4, 5, 6]. They provide such complete coverage of the system that only rarely has access to the MINOS FORTRAN source code been required.

a. MINOS User's Guide

This document is for the original MINOS system, which did not accommodate nonlinear constraints. It is quite thorough and provides some of the mathematical foundations upon which the code is built. It has one section devoted solely to problems with nonlinear objective functions. The input data required by the user is also thoroughly documented with detailed coverage of the options available in the SPECS file as well as some description of the "MPS Format" used by the MPS file. The

various input/output options involving the basis and solution files are addressed in great detail. A complete breakdown of the iteration log output is given to allow the user to trace solution trajectory, if required. There is a good deal of coverage of hardware-dependent matters, since there are some subroutines that are computer manufacturer or machine dependent. Example problems are also given with complete implementation advice.

b. MINOS/AUGMENTED User's Manual

This manual serves as an adjunct to (not a replacement for) the User's Guide. It is intended primarily to update the User's Guide for solving problems with nonlinear constraints, but contains additional information as well. Some mathematical overview is given for the nonlinear constraint case, as well as the necessary additional commands in the SPECS file and the additional subroutine (CALCON) required. There is some alteration of the syntax and keyword meanings and options from the (previously described) MINOS User's Guide, and the coverage is not complete; therefore, both documents must be used for nonlinearly constrained problems. MINOS can most readily be applied by synthesizing both documents to create a "driver template" (such as those used by the X System) for input files and function subroutines; this shows the user most available options, and references those that cannot readily fit in the "template" format. The MINOS/AUGMENTED Manual also contains sample problems with complete input and output listings which provide a useful guide to the new user.

c. MINOS Distribution Documentation

This document serves as a cover letter for the distribution of the code via magnetic tape. It discusses in detail the machine-specific requirements for successful installation of MINOS on the user's computer system. MINOS is currently available with machine-dependent routines tailored for Burroughs, CDC, DEC, Honeywell, IBM, and Univac computers. The documentation also describes a procedure to run test problems on the distributed tape which allow a complete check of the code. Problems 6 and 11 of this study are included as test problems on the MINOS tape. Some minor changes that have been made to the system since the publication of the two previous manuals are also presented.

4. Implementation

The first step in problem solution is one of proper problem formulation. The need for understanding the problem, determining proper scaling, and establishing sensible ranges and bounds cannot be overemphasized and will be discussed more fully in Section III. Ignoring these steps in haste will invariably haunt the analyst in the long run.

a. CALCFG

Subroutine CALCFG provides MINOS with explicit calculations of the nonlinear objective function and its gradient. There is an indicator variable in the calling arguments that indicates the first, last, or some intermediate call to the subroutine, providing it an opportunity to make intiial or final calculations if required.

MINOS allows substitution of numerical difference approximation for the analytical gradient of the objective function.

b. CALCON

Subroutine CALCON serves the same purpose for the nonlinear constraints as does CALCFG for the objective function. However, the option for differencing of the constraints in lieu of analytic gradients is not presently available. The gradient functions must be explicitly provided.

c. SPECS File

The SPECS file consists of 80-column, card-image records that provide MINOS with the parameters of each problem and allow the user to set virtually every control parameter within the code. Each of the card records contains (in free format) a sequence of items that includes a first "keyword," an optional second "keyword," and a number representing the value of the parameter identified by the keywords.

d. MPS File

Constraint matrix data and labels for constraints and variables are provided to MINOS via an MPS file, which (as the name implies) uses standard "MPS Format" [Ref. 3]. In nonlinear problems, the only restrictions are that the nonlinear variables must occur in the left-most columns of the constraint matrix, and the non-linear constraints must be the top-most rows of the matrix. Additionally, an optional INITIAL bounds set may be specified which assigns initial values (if available) to the nonlinear variables in the problem.

The need to state a problem formulation with proper scale for both functions and variables cannot be overemphasized.

e. Basis Files

MINOS provides four different methods for loading or saving basis representations. These options are invoked by use of records in the SPECS file and allow the user to save current values/solutions and reload them at a later time using a number of alternatives. Because of the relatively small size of the problems in this study, these options were not extensively used, but appear to be well thought out and of great potential value in working with large real-life problems.

f. Machine Dependent Routines

MINOS was developed on IBM computer systems, and although every attempt was made to make the code as portable as possible using FORTRAN, some machine dependent details must be changed in order to install the code on other than an IBM machine.

5. Output

MINOS provides a great deal of flexibility in the detail of its output through several SPECS file options as well as the opportunity for using a tailored report writer.

a. Print Level

The most direct control of output is via the PRINT LEVEL option of the SPECS file. This command, which can be thought of as a five-digit binary variable, allows the user to individually select any one or all of the following:

- invert statistics;
- the value of nonlinear variables;
- the Lagrange-multiplier estimates for the nonlinear constraints;

. the values of the nonlinear constraint functions;

. the Jacobian matrix.

In addition, the LOG FREQUENCY command controls how often information is printed from the iteration log. This informs the user at specified intervals of the status of the optimization effort.

b. Solution File

The solution file can be printed on any device (determined by a SPECS file entry) and is a static preformatted report. It is also designed to be subsequently read from disk by a self-contained program which extracts and saves the values required by the user. This allows the user to make general use of the solution without first tailoring the entire output file or modifying the basic code with a tailored report writer.

6. Debugging

In general, MINOS is an easy code to work with as long as the user can interpret the iteration log and ancillary printouts in the more complex cases.

a. Gradient Checks

The requirement for explicitly calculating gradients for the constraints, and preferably for the objective function as well, is a nuisance at best; but it can prove to be helpful in the early debugging phase of a complex problem. MINOS checks the analytic gradient, comparing the function and gradient information provided by the user for consistency. This check normally uncovers errors of coding and/or calculus, or gives the user at least some hope that the provided functions and gradients are correct.

b. Error Statements

Although the diagnostic statements provided by MINOS are explicit and normally lead directly to an error, a few are not documented in the support literature at this time and can lead to some confusion. Consulting the source code is of some help, since it is very well documented for the technically qualified reader. However, not even the source code review has been able to pinpoint the cause of all errors and has forced the user to occasionally resort to more traditional FORTRAN error-checking techniques.

B. X SYSTEM

XS has been developed since 1974 [Ref. 14] as a general-purpose optimization system of advanced design which serves both as a prototype testbed for research and as the fundamental computational foundation of many application packages utilizing optimization.

XS is designed to solve large-scale optimization problems, with special emphasis on mixed integer models. Decomposition features are of premier importance to XS at an aggregate level of detail. However, the embedded linear programming module has received the most design effort--it is the heart of XS and exhibits many singular features including:

- hyper-sparse data representation [e.g., Ref. 15];
- complete, constructive degeneracy resolution [e.g., Ref. 16: pp. 1-34];
- basis factorization [e.g., Ref. 17: pp. 91-110];
- elastic range constraints [Ref. 14].

The nonlinear feature of the X System is designed for use with large-scale models exhibiting some functional nonlinearity (non-separable,

nonlinear objective and constraints are admitted). However, XS is designed to efficiently solve models for which nonlinearity is relatively mild, hopefully involving a subset of model variables and constraints. Large refining models provide an ideal nonlinear paradigm in this context.

Design criteria for XS require that all other features be supported simultaneously with the nonlinear feature (e.g., see Problem 12 with generalized upper bound (GUB) factorization and mixed integer (MIP) variables in addition to nonlinear features). This inflexible rule derives both from the wide variety of production applications currently using XS, as well as from the research philosophy of the developers.

The basic nonlinear feature of XS is designed to encourage reliable, error-free problem input and solution, minimizing the total response time including problem preparation, debug, tuning, and interpretation of output. Accordingly, gradient functions are not required, nor are they even suggested as an attractive option to any but the most skilled user. Long experience has shown that coordination of gradient functions with problem functions reliably inflicts more frustration than all other steps in solving nonlinear models! The X System is designed for robust performance with automatic numerical difference approximation for all functions.

Another notable nonlinear feature of XS is the pervasive use of robust problem-independent algorithm controls, such as a dynamic "trust region" for approximation of the problem functions, a conservative ray search, and heuristic internal tolerances to stabilize performance in the presence of "irregularities" such as function discontinuities (common to many models, especially those that are new and those that employ spline approximations [see Section III.B] from imprecise data observations).

Also, use of the elastic ranges invites elegant formal incorporation of internal estimates for approximate data, prioritizing of constraints, and various relaxation methods.

XS completely consists of open FORTRAN subroutines. FORTRAN IV (H Extended) [Ref. 2] is the implementation dialect and an IBM 3033 the host computer. XS is installed and operating on other systems with compatible arithmetic architecture, and the FORTRAN subset actually used is widely accepted, even by FORTRAN 77 [Ref. 18]. In-core, out-of-core procedures, if invoked, require the most installation-specific preparation.

Problem input may include any combination of explicit function definitions, sparse row and/or sparse column generators, and MPS Format. No tuning parameters, gradients or other problem-specific data are required other than some adequate (or default) estimate of sheer problem dimensions (rows and columns). An example of the minimum information necessary to invoke the algorithm for a sample problem (Problem 9) is provided in Figure 4.

Input may use any of several standard FORTRAN subroutine "templates" which require only rudimentary programming skill. Alternate interface procedures may exercise XS directly for extremely challenging problems. All relevant tuning defaults and tolerances are available for user override via a simple menu in FORTRAN subroutine form.

XS employs an (NLP) technique which generates a sequence of local linear programs (LP), each from previous solution and gradient estimates. Each local LP determines a search direction based on a first-order local approximation of the problem functions and on the dual variable bounds (elastic range penalties), forming a piecewise linear Lagrangian gradient

Formulation:

$$\text{maximize} \quad y_1 \cdot y_2$$

$$\text{subject to} \quad \frac{y_1^2}{(30)^2} + \frac{y_2^2}{(23)^2} = 1$$

$$y_1, y_2 \geq 0$$

Implementation:

```
M    = 1                                (number of constraints)
N    = 2                                  (number of variables)
G(1) = Y(1)**2/900.0D0 + Y(2)**2/529.0D0   (constraint)
GO = Y(1) * Y(2)                          (objective function)
```

Fig. 4. X System Problem Generation in Explicit Mode

and investigating this general search direction with a local, closed-interval ray search.

XS reliably solves NLP's with excellent efficiency. More important, XS can be tuned for particular classes of problems to provide unparalleled solution speed. However, XS employs a first-order descent method, and one would expect that convergence to more than 3-4 decimal place objective function tolerance will occasionally require disproportionate effort. The designers have consciously opted for fast, easy, absolutely reliable convergence to 3-place precision, a tolerance frequently exceeding coefficient data resolution. Although much better solution resolution is

desirable, and routinely achieved for amenable problems, it is not guaranteed by XS without some experience (tuning).

For those cases in which extreme solution tolerance is demanded, a second-order variation is available which augments the explicit constraints with the first-order stationary conditions. This method has produced excellent results [Ref. 19], but the additional problem set-up time is seldom economically justified in lieu of simply running default first-order XS a bit longer. (This represents a classic paradox in evaluating algorithms and their actual application.) Given the additional set-up effort, robust hybrid first- and second-order descent algorithms are easily implemented by use of the constraint penalties as descent forcing functions.

1. Algorithm

The X System requires that (NLP) be restated elastically:

$$\text{(ENLP)} \quad \text{minimize} \quad g^0(y) = f(y) + \underline{z}a + \bar{z}r$$

$$\text{subject to} \quad g(y) + s + a - r = \bar{r}$$

$$\underline{b} \leq y \leq \bar{b}$$

$$o \leq s \leq \bar{r} - \underline{r}$$

$$a, r \geq 0$$

or, in an equivalent Lagrangian form:

$$\text{minimize} \quad f(y) + z^T(g(y) - r)$$

$$\text{subject to} \quad \underline{b} \leq y \leq \bar{b}$$

$$\text{with} \quad z_i = \begin{cases} \bar{z}_i & ; \text{ if } g_i(y) > r = \bar{r}_i \\ -\underline{z}_i & ; \text{ if } g_i(y) < r = \underline{r}_i \\ 0 & ; \text{ otherwise .} \end{cases}$$

We see immediately that $\underline{z}$, $\bar{z}$ are interpreted both as constraint penalties and as linear dual bounds, and that convexity is preserved with $\underline{z}, \bar{z} \geq 0$.

Given some solution for (ENLP),

$$\underline{b} \leq y^0 \leq \bar{b} ,$$

a local elastic linear program is generated:

$$\begin{aligned} \text{(LELP)} \quad & \text{minimize} \quad \nabla g^0(y^0)\Delta y + z^T(\nabla g_i(y^0)\Delta y - r) \\ & \text{subject to} \quad \underset{\sim}{b} \leq \Delta y \leq \tilde{b} \end{aligned}$$

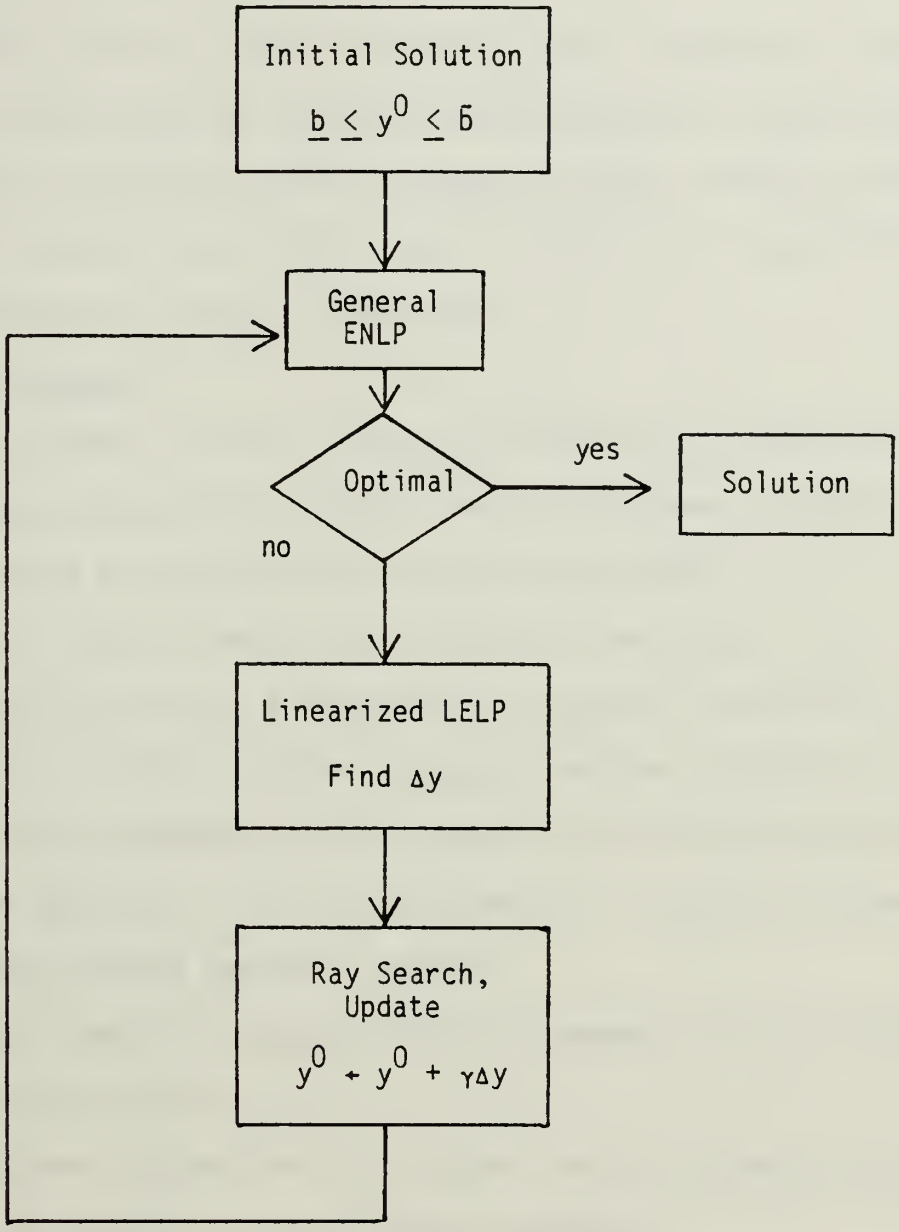
$$\text{with} \qquad z_i = \begin{cases} \tilde{z}_i & ; \text{ if } \nabla g_i(y^0)\Delta y > r = \bar{r}_i - g_i(y^0) \\ -\underset{\sim}{z}_i & ; \text{ if } \nabla g_i(y^0)\Delta y < r = \underline{r}_i - g_i(y^0) \\ 0 & ; \text{ otherwise} \end{cases}$$

where $\underset{\sim}{b}$, $\tilde{b}$ are local bounds derived from $\underline{b}$, $\bar{b}$ and enforced as a local "trust region" over which the functions g are approximated acceptably by first-order Taylor series and $\underset{\sim}{z}$, $\tilde{z}$ are local penalties derived from $\underline{z}$, $\bar{z}$ (and optionally other artifacts of solution).

The complete algorithm performs a ray search on the piecewise linear convex (for convex functions g) descent direction Δy, updates y^0, and generates another local elastic program if terminal conditions are not satisfied (see Figure 5). In particular, descent is assumed if:

a). The latest improvement in the objective function is greater than a (specified) tolerance, and

b). The maximum number of (specified) iterations has not been exceeded.



Note: Integer detail ommited from NLP solution process for clarity.

Fig. 5. X System Iteration Flowchart

Convergence is assumed if there is no descent for a (specified) number of iterations.

At each iteration, the local trust region is determined dynamically based upon (specified) minimum and maximum desired step-length (in y-units). It is increased in size if descent has been achieved, decreased otherwise. In addition, each local bound is modified by a (specified) factor if oscillatory behavior is observed.

2. Code Structure

The X System is primarily designed and used as an "open subroutine" system, but can be run "stand-alone" for problems of specified structure. Figure 6 shows the module organization of XS.

Default tuning parameters are defined in the NUCLEUS. For problems of given structure, a "template" is prepared consisting of two entry points. PROB initializes problem generators and specifies any non-default initial parameters; FGE provides function values as required by the problem generators, and can also dynamically change any parameter.

Standard problem templates include:

- Explicit Mode (FGE provides explicit statements of the problem functions in terms of y);
- Sparse Mode (FGE provides first-order function information only once, and computes nonlinear terms on demand);
- MPS Mode (Sparse Mode with MPS input);
- Super-sparse Mode (PROB and FGE communicate directly with internal XS data structures).

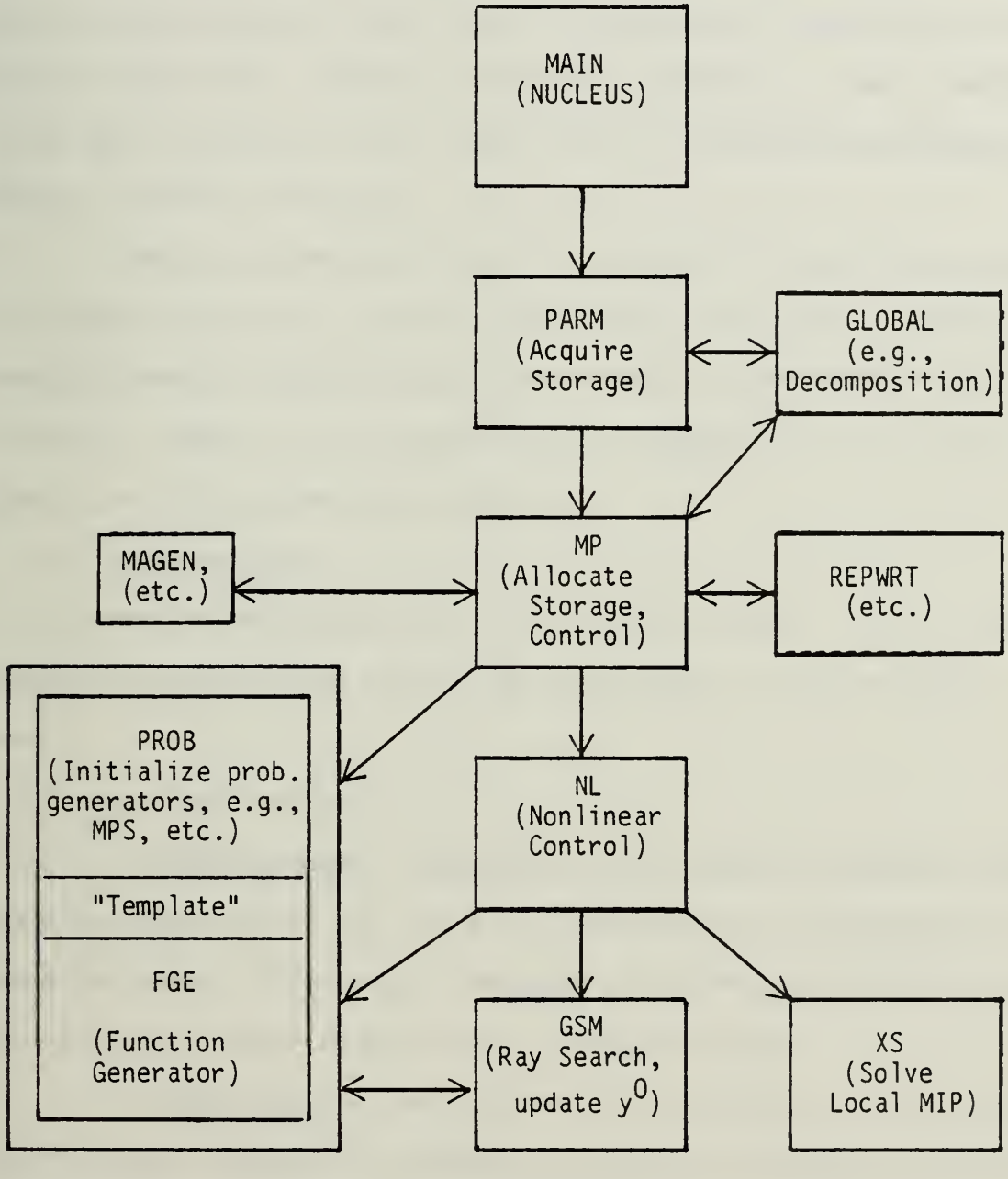


Fig. 6. X System Module Structure

3. Documentation

User documentation for XS is terse. A brief summary of the system and examples of LP, MIP, and NLP formulations, templates, set-ups, and solutions are given [Ref. 20]. The mathematical foundations are set forth in [Ref. 21]. Finally, the designers provide an internal maintenance specification and logic manual with full current implementation details and data structures.

Complete documentation for XS is prepared for particular production applications [e.g., Ref. 22, 23]. Also, user-friendly interactive templates permit "classroom use" of XS in real-time with very little training. However, full documentation for prototype XS is not likely to appear until the system stops improving.

4. Implementation

Problem preparation for XS is straight-forward. First, a template should be selected which matches the requirements of the problem at hand.

a. Mode Selection

Explicit Mode -- Adequate for small and intermediate-sized problems, very easy to use, but a bit less efficient than other, more demanding modes. Efficiency is enhanced if nonlinear constraints and variables are identified for XS in a contiguous block.

Sparse Mode -- Desirable for large problems and relatively easy to use. Efficiency is enhanced if non-null coordinates of nonlinear constraints and variables are identified.

MPS Mode -- Required for problems from outside sources expressed in MPS Format. This mode also provides one means of introducing

labels for functions and variables using proprietary matrix generators, identifying non-null coordinates of nonlinear constraints and variables, and providing a PENALTIES section in an easy format. However, processing MPS Format files is relatively expensive, and editing requires another file with revision instructions.

Super-sparse Mode -- Greatest efficiency and risk. Mandated only by huge problems, complex generators, or other uncommon model curiosity.

Once the proper template is selected, the following are assembled, as required.

b. PROB

Subroutine PROB defines initial conditions (connects MPS files as necessary) and sets up problem generators. Default tuning parameters may also be redefined.

c. FGE

Subroutine FGE generates function values for a given solution. The calling discipline provides a single argument which invokes complete function evaluations, only nonlinear function evaluation, only non-null coordinate function evaluation, or only non-null nonlinear function evaluation, depending upon the template adopted. Except for super-sparse mode, the solution, base (partially evaluated) function values and complete function values are all available in precisely the organization (order) specified by the user. In super-sparse mode, basis factorization and permutation vectors may be exercised directly by the user, as well (not suggested for the timid!). Tuning parameters may be redefined dynamically, as may any problem element.

d. MPS File

This is required only in MPS mode. A PENALTIES section may be used, as may an associated MPSEDIT file. Both files are in card image form with 80-character, fixed-length records.

e. SCRATCH Files (problem independent options)

These files are respectively required only for in-core, out-of-core use, for online storage of incumbent solutions in non-convex and decomposed problems, and for efficient reinversion of the basis. Unformatted FORTRAN Input/Output is used with the attendant file definition requirements.

f. CRASH File (problem independent options)

This is required for basis crash options, for basis save options, and for efficient reinversion. Unformatted FORTRAN Input/Output is used.

g. Machine Dependent Routines (problem independent options)

For IBM architecture, XS may use a high-resolution computation timer, an integer overflow arithmetic hardware interrupt trap, and a dynamic (GETMAIN) storage allocation. These routines are coded in assembly language.

5. Output

Standard output includes various (specified) levels of detail, ranging from absolutely nothing to complete logs of each primitive operation within each segment of each minor iteration. Frequently useful reports include:

Input Summary -- by row and column giving labels (if any), basis factorization status, selection status, non-null element counts, maximum

magnitude and range of first-order elements, range and bound status and scale, and penalty scale.

Set-up Summary -- with tuning parameters, dimension keys, host environment, XS vintage, and intended problem template.

Major Iteration Log -- with complete time and pivots, solution value (with and without penalty terms), ray search resolution, local gain, trust region size, and descent condition.

Minor Iteration Log -- (optional frequency) with linear information as in Major Iteration Log.

Exception Log -- with basis crash and stability recovery diagnostics.

Solution Summary -- by row and column giving labels (if any) and complete elastic primal, or dual solution (or edited selected subsets).

Final Summary -- with total pivots and pivots per iteration, total function calls and calls per iteration required by ray search, integer, and differencing operations, respectively, and total compute time.

Solution File -- for automated real-time solution analysis [e.g., Ref. 24], or off-line report writing.

In addition, a report writing template is provided for use in customizing output formats and report content.

6. Debugging

Nonlinear models fail most frequently in just three ways: programming errors in the problem representation fail to mate it compatibly with calling modules and data structures, or model errors for which a correctly coded problem representation generates an incorrect model, or

solution errors yield undesired stationary points from otherwise correct models.

Some programming errors are almost completely undetectable by an open FORTRAN subroutine (e.g., subscript range errors, argument list omissions, etc.). Provision of standard templates is the best insurance measure for these perennial annoyances. Symptoms of such difficulties are usually severe, immediate, and fatal. A change of host language would ameliorate this situation, but benchmarks of FORTRAN competitors have not yet revealed a candidate with sufficient execution efficiency for economic use with XS.

Since XS uses no analytic gradients (a profound advantage in model debugging), one need usually merely verify that a given solution y^0 produces function values correctly. The set-up and solution summaries of XS usually diagnose further trouble very quickly--the elastic range violations reported in any solution focus attention on likely candidate functions for further inspection.

Of course, XS can minimize an objective function intended for maximization, and can produce stationary points for completely incorrect formulations and implementations; however, these are intellectual issues for the modeller to ponder more than bugs in the classic sense. No debugging aid can guarantee reliable solution of a model which is not understood by the user.

Interpretation of terminal solutions sometimes reveals that a local stationary point has been selected. XS provides an alternate procedure for problems suspected of such behavior, whereby the objective function is included with the constraints, and its penalties used to

enforce successive level set descent at each major iteration (e.g., Problem 4 for which XS has found three stationary points). Non-convex problems usually also require that the ray search be tuned for extra effort.

Some tuning advice is always a valuable part of solution debugging. For variables, XS uses a zero level and infinity value which may be altered. The trust region is used when unruly nonlinear function behavior (misleading first-order approximation) is predicted, or as an additional safeguard which costs little. A distortion factor is provided by which oscillatory behavior detected in successive solutions is damped. (This is no substitute for conjugate directions or second-order information, but serves well in practice as an effective heuristic).

A zero tolerance is used for functions, and may be altered. XS also requires an objective function tolerance for nonlinear convergence (and for integer convergence), as well as penalties for violation of each constraint range. Thus, the modeller must express a priori in terms of a single objective function the consequences of violating feasibility and optimality.

Default values for all tuning parameters are necessarily scale dependent and require reasonably well-scaled models for reliable performance. In this sense, the X system is rather unforgiving of users who do not express their models completely and carefully.

III. EXPERIMENTAL METHOD AND RESULTS

A. METHOD

In preparation for this study, a copy of the documentation [Ref. 4, 5, 6] and code for MINOS were received from Saunders (circa May 1980, via G. Brown from a Vienna meeting with Saunders). The code was received on magnetic tape which contained the system source code (both machine dependent and portable), test problem source code, assembler language source code, object module, and executable load module. The machine dependent routines, assembler code, and the object modules were all compatible for the IBM system on which this test was run. The OPTIMIZE (3) load module from the tape was not used, but replaced with one compiled at the Naval Postgraduate School with the OPTIMIZE (2) option [Ref. 2]. (OPTIMIZE (3) is not available at the Naval Postgraduate School at this writing.)

The X-system was already in place on the Naval Postgraduate School computer and was used in situ. Its working "documentation" [Ref. 20] was also available.

Because of the complete MINOS documentaion and the availability of several test problems already set up for immediate execution, all problems were initially prepared for MINOS and then converted to XS (except as noted). This procedure required a greater initial outlay of effort (See Section II.B) because of MINOS' requirement for gradient information not required by XS; however, the consistency checks make internally by MINOS of the function and gradient routines served as a troubleshooting tool to

ensure the functions had been correctly coded. As a result, the conversion of each problem from MINOS to XS took less than an hour in most cases using the computer system file editor, once the problems were successfully run with MINOS. The converse was not true, however, as the gradient functions required considerable time and effort to derive and implement for MINOS.

Both systems are presently being developed, supported, and used by their authors on IBM 3033's. Therefore, neither system has been given an advantage by the computer used to support the evaluation.

B. PROBLEM FORMULATION COMMENTS

Although "easy" problems can be solved with brute force by either system, large or tricky problems require intelligent problem formulation for either system to be successful. Among the most vital guidelines to consider are the following.

The analyst must UNDERSTAND THE NONLINEAR PROBLEM in order to successfully solve it (or in order to assimilate the answer). This can be a major hurdle when solving someone else's nonlinear problem formulation or test problem if inadequate documentation exists to set up the problem in a format that is compatible with the optimization code being used. Although true for both optimization systems, in the case of XS this is absolutely vital. A good deal of the power of XS lies in the use of its "elastic" penalties on the constraints. If the problem is not fully understood, these penalties cannot be set intelligently and at best the power of XS is diminished, because the penalties require formal transformation by the modeller of infeasibilities to units of the objective function. At worst, XS may not be able to solve the user-stated problem

at all since it views the elastic formulation as "the problem." This can be a major hurdle when solving someone else's nonlinear problem formulation or test problem if inadequate documentation exists to set up the problem in a format that is compatible with the optimization code being used.

There is a tendency in many problem formulations to overlook points within the allowable variable bounds that generate singularities in the problem functions. By Murphy's Law[1] and O'Toole's Corollary,[2] it is a virtual certainty that any code will attempt to evaluate the functions at a singularity if proper precautions are not taken by the analyst. These precautions may be in the form of restricted bounds on the variables that were not included in the original problem specification or in coding within the function generator.

For example, in the region of a singularity a spline function may be substituted for the function of interest at some point such that:

- Endpoints are satisfied by the composite function;
- The composite function is continuous;
- The composite function has a continuous derivative;
- The composite function is a "reasonably" close approximation of the function and its gradient (e.g., a "majorizing function").

In the same vein, reasonable bounds for the variables should be specified to prevent excessive run times with either system. Even if the problem has been initially specified with unbounded variables, the intelligent analyst can restrict the problem to reasonable bounds. There are

---

[1] If anything can go wrong, it will . . . at the worst possible moment . . . and in the way so as to maximize the damage.

[2] Murphy was an optimist.

few real-life problems for which an infinite value for a variable makes sense.

Both systems, in fact any useable nonlinear optimization code, are sensitive to scale and require that the analyst formulate the problem in a reasonable manner. In general, the data, variables, and functional values should be as close to 1.0 as possible. Due to efficient specializations of both algorithms for linear (or nearly linear) constraints, it is worth a good deal of effort by the analyst to minimize the nonlinearities in the problem by appropriate transformations. A nonlinear objective function is usually preferable to nonlinear constraints (if a choice exists via problem transformations).

It is mandatory with XS and strongly recommended with MINOS that any initially specified solution lie within the bounds of the variables.

C. PROBLEM DESCRIPTIONS

The first eight of the test problems had been used for a previous study [Ref. 25] and were collected from a variety of sources. The first six problems are listed in Himmelblau [Ref. 26: pp. 395-425] and the original source author/developer for each is listed. The next two are adaptations of an inventory model [Ref. 27] and an entropy model [Ref. 28] which were specifically designed to illustrate real-world problems with few constraints but many independent variables, albeit with a relatively small number of variables compared with the capabilities of the two codes under test. The ninth problem was developed by the author during course work at the Naval Postgraduate School.

Appendix A provides the mathematical problem statement, problem data, starting points, and an initial and final solution from each code

for all problems except Problem 11: the formulation of ETAMACRO provided as a test problem for MINOS by its developers did not exactly match the description of [Ref. 29]. Time considerations precluded resolution of the discrepancies, and although the MINOS test problem formulation was used, neither formulation is presented here.

The remaining four problems were not solved by both codes either due to sheer time constraints, or in the case of Problems 12 and 13, because integer variables cannot be accommodated by MINOS.

The alternate starting points for problems 1-8 were taken from Waterman [Ref. 25: pp. 56-91].

1. Problem 1 (Himmelblau 6)

This problem [Ref. 30[3]] is an example of determining the chemical composition of a complex mixture under conditions of chemical equilibrium. It contains 45 independent variables and 16 linear equality constraints. MINOS returned a solution very quickly, although the solution from the second alternate starting point differed from the first two starting points. XS took a bit longer to reach a solution from all three starting points; however, the objective value was improved over the MINOS solutions and the three solutions agreed with only minor variations.

2. Problem 2 (Himmelblau 4A)

This is also a chemical equilibrium problem which had been redefined in the Himmelblau study from a problem originally formulated and solved by Dantzig, Johnson, and White [Ref. 31: pp. 751-755] and discussed by Bracken and McCormick [Ref. 32: pp. 46-49]. It contains 10 independent variables and 3 nonlinear equality constraints. Both codes

[3]Reference not viewed by author.

obtained essentially identical solutions from all three starting points in about the same amount of time.

3. Problem 3 (Himmelblau 18)

This problem was formulated by the Shell Development Company for the original Colville study [Ref. 33: p. 22] and consisted of 15 independent variables and 5 nonlinear inequality constraints. MINOS returned a slightly better solution, though both codes returned solutions from all three starting points in approximately the same amount of time.

4. Problem 4 (Himmelblau 16)

This problem [Ref. 34[4]] maximizes the area of a hexagon in which the maximum diameter is unity. There are nine independent variables, 13 nonlinear inequality constraints, and a lower bound of zero on one variable. This problem contains many local stationary points and its solution difficulty is not reflected in the problem summary in Section D of this chapter. Both codes required a little tuning in order to avoid local minima. To verify both MINOS and XS solutions, an XS option was invoked which uses an alternate formulation of the problem using the original objective function as a constraint, replacing it with an implicit objective function. (Recall that XS implicitly uses penalties in its objective function.) This constraint value then provides the actual objective function value in the final solution. Although the final results achieved for XS using the original formulation exceeded the alternate method in speed, alternate formulation appears to be very

[4]Reference not viewed by author.

robust and seems to be a useful tool with much to recommend it for XS used on problems containing multiple local starting points.

5. Problem 5 (Himmelblau 20)

This problem [Ref. 35[5]] contains a linear objective function, 24 independent variables, 12 nonlinear equality, two linear equality, and six nonlinear equality constraints. The independent variables also are bounded to positive values. Both codes returned identical solutions from all three starting points, however, XS required more time to reach the solution.

6. Problem 6 (Himmelblau 23--continuous relaxation)

This is a weapon assignment problem formulated by Mylander [Ref. 36[5]] and presented by Bracken and McCormick [Ref. 32: pp. 22-26] with 100 independent variables, a nonlinear objective function, 12 linear constraints and zero lower bounds for all variables. This problem was formulated for XS using the GUB option. Both codes returned approximately equal objective function values from all three starting points, however, the returned variable values were quite diverse, which comfirmed previously reported results [Ref. 25, 26]. XS required somewhat longer to reach the solution.

Since it was suspected that the variability in reported results might be due to the integer nature of the variables in the original formulation which had been approximated with continuous variables by all previous solution attempts, the problem was reformulated with integer (binary) variables using XS and solved again as Problem 12 below.

---

[5]Reference not viewed by author.

7. Problem 7 (Waterman 7)

This problem was adapted by Waterman [Ref. 25] from an inventory model created by Choe and Schrady [Ref. 27: pp. 451-463]. The first 50 variables represent the reorder quantity for 50 inventory items and the next 50 variables represent the reorder points for the same 50 items. The problem contains one linear and one nonlinear inequality constraint and 50 lower bounds on the variables. This was the most difficult problem solved by both codes. MINOS returned approximately the same solutions from all three starting points, however much more time was required from the alternate starting points. MINOS could not achieve a solution from the second starting point with the conjugate gradient option, but achieved much quicker solutions from the remaining two points. XS did not return as good a solution from any of the starting points and also required more time to solve the problem.

8. Problem 8 (Waterman 8)

This problem was adapted from an entropy model proposed by Scott [Ref. 28: pp. 204-211]. The nodes depict 46 population centers connected by a transportation network, represented by the connecting arcs. Using a congestion cost function, the model yields an equilibrium solution that identifies node populations as entropic functions of the total cost of transportation to a central place of work. The problem has 46 independent variables, one nonlinear inequality and one linear equality constraint, and 46 lower bounds on the variables. MINOS returned the same solutions from all three starting points. Solution times for both systems changed significantly with starting points.

9. Problem 9 (Dean 9--Game)

This is an example of a small problem developed from a classroom exercise to serve as an example of how quickly problems of a very basic nature can be solved using the codes as time-saving tools. The problem has one nonlinear equality constraint and two independent variables. Both codes found the solution from all three starting points in comparable amounts of time. It took less than an hour (clock time) from start of initial problem formulation until solutions were available from both codes.

10. Problem 10 (Dean--Sortie)

This problem is from current work being done by the U.S. Air Force based on a model done by Clasen, Graves, and Lu [Ref. 37]. It contains 81 linear constraints and 793 independent variables. Because of time constraints and an unsuitable data format for direct input to MINOS, the problem was only solved using XS. The GUB feature of XS was also used on this problem. The results are given to provide a measure of comparison of this relatively large problem with the other smaller test problems. Also, the run time for this problem on the host computer using the Air Force's present, but dated, solution algorithm is 321 seconds.

11. Problem 11 (Dean--ETAMACRO)

This is Manne's energy problem ETAMACRO [Ref. 29: pp. 1-45] and is included on the MINOS distribution tape as a test problem. It contains 401 linear constraints and 688 independent variables. Because the model was formulated in close conjunction with the developers of MINOS, the formulation is very compatible with that code's structure. However, it

was not suitably formulated to exploit the elastic features of XS and although the problem was solved in its linear form by XS, time constraints precluded a complete reformulation which would have been necessary in order to solve the problem efficiently on XS. It is presented here because of its relatively large size in comparison to the other test problems, in order to give some indication of MINOS' performance on large problems.

12. Problem 12 (Himmelblau 23--Integer Nonlinear)

This is a reformulation of Problem 6 with integer (binary) variables instead of the continuous approximations used in Problem 6. XS's GUB option was also exercised on this problem. Because of the integer nature of the variables, the problem was only run on XS. this problem has never before been solved as a nonlinear integer model.

13. Problem 13 (Dean--Integer Nonlinear)

This problem [Ref. 38: pp. 519-536] was submitted by Schmit to Graves and is an example of a current engineering design problem that is highly nonlinear and contains integer variables. It has 48 nonlinear inequality and three linear inequality constraints and 12 independent variables, of which 8 are integer (binary). Because of the integer variables, the problem was solved only with XS, but is listed as an example of a nonlinear integer problem solution by XS.

D. PROBLEM SUMMARY

This problem summary lists results for MINOS using the default (on small problems) quasi-Newton method as well as the conjugate gradient option normally used only for large problems. This is done to present a complete comparison of the two codes using their respective large-scale algorithms. XS does not provide an automatic second-order descent method

option (although [Ref. 19] presents the groundwork for such an option without the "elastic" framework) since it would be of little use with the large problems for which XS is designed.

	Primary Solution	Alternate Solution	Alternate Solution
Problem 1 (Himmelblau 6)			
MINOS (quasi-Newton)			
Obj. Fn.	-1895.4	-1895.9	-1906.5
Linearizations	1	1	1
Pivots	49	50	49
Fn. Evals.	16	15	15
CPU time	0.07	0.06	0.06
(Conj. Grad.)			
Obj. Fn.	-1895.4	-1895.9	-1906.5
Linearizations	1	1	1
Pivots	49	50	49
Fn. Evals.	16	15	15
CPU time	0.09	0.07	0.07
XS (compact template)			
Obj. Fn.	-1910.0	-1910.0	-1910.0
Linearizations	3	3	3
Pivots	154	130	145
Fn. Evals.	151	150	150
CPU time	0.39	0.36	0.41

	Primary Solution	Alternate Solution	Alternate Solution
Problem 2 (Himmelblau 4A)			
MINOS (quasi-Newton)			
Obj. Fn.	-47.761	-47.761	-47.761
Linearizations	16	14	24
Pivots	208	126	267
Fn. Evals.	1016	608	1132
CPU time	1.11	0.67	1.31
(Conj. Grad.)			
Obj. Fn.	-47.761	-47.761	-47.761
Linearizations	16	14	24
Pivots	501	281	496
Fn. Evals.	3110	1620	3194
CPU time	2.68	1.46	2.70
XS (compact template)			
Obj. Fn.	-47.766	-47.762	-47.761
Linearizations	41	54	45
Pivots	397	469	456
Fn. Evals.	642	816	716
CPU time	0.80	0.98	0.82
Problem 3 (Himmelblau 18)			
MINOS (quasi-Newton)			
Obj. Fn.	-32.349	-32.349	-32.349
Linearizations	7	8	9
Pivots	76	61	93
Fn. Evals.	329	264	403
CPU time	0.32	0.29	0.43
(Conj. Grad.)			
Obj. Fn.	-32.349	-32.349	-32.349
Linearizations	8	12	9
Pivots	121	346	444
Fn. Evals.	626	2110	2525
CPU time	0.51	1.60	1.93
XS (compact template)			
Obj. Fn.	-34.228	-38.651	-33.546
Linearizations	7	9	18
Pivots	100	102	177
Fn. Evals.	153	191	393
CPU time	0.17	0.20	0.37

	Primary Solution	Alternate Solution	Alternate Solution
Problem 4 (Himmelblau 16)			
MINOS (quasi-Newton)			
Obj. Fn.	-0.86603	-0.86603	-0.86603
Linearizations	7	9	20
Pivots	52	87	168
Fn. Evals.	207	317	688
CPU time	0.29	0.42	0.96
(Conj. Grad.)			
Obj. Fn.	-0.86603	-0.86603	-0.86603
Linearizations	7	9	20
Pivots	78	120	324
Fn. Evals.	370	551	1698
CPU time	0.39	0.55	1.67
XS (compact template)			
Obj. Fn.	-0.86606	-0.86605	-0.86606
Linearizations	28	32	29
Pivots	356	340	356
Fn. Evals.	413	551	427
CPU time	0.40	0.51	0.41
Problem 5 (Himmelblau 20)			
MINOS (quasi-Newton)			
Obj. Fn.	0.05566	0.05566	0.05566
Linearizations	6	11	7
Pivots	52	104	62
Fn. Evals.	48	129	85
CPU time	0.63	1.28	0.91
(Conj. Grad.)			
Obj. Fn.	0.05566	0.05566	0.05566
Linearizations	6	11	7
Pivots	56	122	88
Fn. Evals.	61	185	149
CPU time	0.76	1.57	1.32
XS (compact template)			
Obj. Fn.	0.05566	0.05566	0.05566
Linearizations	8	13	11
Pivots	476	628	514
Fn. Evals.	444	686	591
CPU time	1.61	2.45	1.78

	Primary Solution	Alternate Solution	Alternate Solution
Problem 6 (Himmelblau 23--continuous relaxation)			
MINOS (quasi-Newton)			
Obj. Fn.	-1735.6	-1735.6	-1735.6
Linearizations	1	1	1
Pivots	145	159	159
Fn. Evals.	311	343	343
CPU time	1.75	1.96	1.96
(Conj. Grad.)			
Obj. Fn.	-1735.6	-1735.6	-1735.6
Linearizations	1	1	1
Pivots	243	297	297
Fn. Evals.	630	742	742
CPU time	2.67	3.21	3.22
XS (compact template)			
Obj. Fn.	-1734.5	-1735.0	-1734.6
Linearizations	38	31	33
Pivots	2424	2194	2390
Fn. Evals.	4068	3333	3542
CPU time	3.85	3.22	3.51
Problem 7 (Waterman 7)			
MINOS (quasi-Newton)			
Obj. Fn.	80.744	80.727	80.727
Linearizations	11	33	21
Pivots	373	1227	784
Fn. Evals.	2087	6927	4231
CPU time	28.03	70.55	54.67
(Conj. Grad.)			
Obj. Fn.	80.744		80.727
Linearizations	19	No	15
Pivots	662	Solution	507
Fn. Evals.	4657		4041
CPU time	14.64		12.53
XS (compact template)			
Obj. Fn.	85.231	80.726	90.604
Linearizations	54	144	117
Pivots	4489	14088	9207
Fn. Evals.	5915	15283	12798
CPU time	26.31	67.68	57.32

	Primary Solution	Alternate Solution	Alternate Solution
Problem 8 (Waterman 8)			
MINOS quasi-Newton)			
Obj. Fn.	-3.4685	-3.4685	-3.4685
Linearizations	6	9	8
Pivots	148	231	146
Fn. Evals.	718	1163	715
CPU time	4.09	6.41	4.06
(Conj. Grad.)			
Obj. Fn.	-3.4685	-3.4685	-3.4685
Linearizations	6	8	5
Pivots	163	240	142
Fn. Evals.	991	1417	884
CPU time	2.79	4.04	2.48
XS (compact template)			
Obj. Fn.	-3.4519	-3.4689	-3.4643
Linearizations	13	40	12
Pivots	787	1842	676
Fn. Evals.	1225	3871	1174
CPU time	3.03	8.02	2.57
Problem 9 (Dean 9--Game)			
MINOS (quasi-Newton)			
Obj. Fn.	345.00	345.00	345.00
Linearizations	6	5	5
Pivots	12	8	6
Fn. Evals.	52	39	37
CPU time	0.05	0.03	0.03
(Conj. Grad.)			
Obj. Fn.	345.00	345.00	345.00
Linearizations	6	5	5
Pivots	12	8	6
Fn. Evals.	52	39	37
CPU time	0.04	0.03	0.02
XS (compact template)			
Obj. Fn.	344.94	342.59	345.01
Linearizations	33	3	10
Pivots	63	3	14
Fn. Evals.	293	44	87
CPU time	0.15	0.01	0.05

	Primary Solution
Problem 10 (Dean 10--Sortie)	
XS (super-sparse)	
Obj. Fn.	2.0087D+6
Linearizations	3
Pivots	977
Fn. Evals.	2393
CPU time	8.60
Problem 11 (Dean 11--ETAMACRO)	
MINOS (quasi-Newton)	
Obj. Fn.	1337.7
Linearizations	1
Pivots	1331
Fn. Evals.	1926
CPU time	41.36
(Conj. Grad.)	
Obj. Fn.	1337.7
Linearizations	1
Pivots	32640
Fn. Evals.	68868
CPU time	1028.05
Problem 12 (Himmelblau 23--Integer Nonlinear)	
XS (compact template)	
Obj. Fn.	-1734.2
Linearizations	30
Integer Linear.	3
Pivots	285
Fn. Evals.	211
CPU time	4.41
Problem 13 (Dean 13-- Integer Nonlinear)	
XS (compact template)	
Obj. Fn.	0.12000
Linearizations	6
Integer Linear.	3
Pivots	63
Fn. Evals.	145
CPU time	0.55

IV. CONCLUSIONS

A. ALGORITHM CAPABILITIES

1. Type of Problems

MINOS is capable of reliably solving both the general (LP) and (NLP) problems with any combination of linear and nonlinear constraints, but cannot accommodate integer variables.

XS has the same capabilities as MINOS with the addition of the use of integer variables. XS can also employ decomposition and basis factorization.

2. Growth Possibilities

There is no inherent maximum problem size for either MINOS or XS. For sheer capacity, they are both limited by the computer memory requirements for their working arrays. However, as noted in Section II.A.1 for MINOS, in large problems (when the number of superbasic variables exceeds 100 or 200) the shift to the conjugate gradient algorithm, which consumes less memory, results in a significant decrease in the theoretical rate of convegence of the algorithm [Ref. 5: p. 10].

B. CPU TIME

MINOS was a bit quicker in solving many of the problems of this study. While XS yields 2-3 decimal place precision in the objective function faster than MINOS, MINOS usually prevails in 3-5 place efficiency (even when using first-order conjugate gradient options).

C. STORAGE REQUIREMENTS

For linear programs containing m general constraints, roughly 100(m)-bytes of memory are required for workspace by MINOS. If there are many nonlinear variables, additional memory may be required. This workspace size may be adjusted by changing the size of one array in the main program for MINOS or by a non-FORTRAN routine that allocates storage at run-time. The choice of method is machine-dependent and guidelines are provided to the user in the documentation [Ref. 4].

XS, used strictly in-core, requires a region of approximately 56(MN) + 8(NR) + 200K-bytes where MN is the total rows + cols, and NR is the size of the distinct real value pool. Storage requirements for nonlinear problems known to this writer are not a significant consideration for XS, or for MINOS.

D. NUMBER OF ITERATIONS

Each major iteration of MINOS creates a local linearization of the nonlinear program, and then solves it after addition of a quadratic (augmented Lagrangian) objective function. XS simply solves local linearizations (with augmentation of the linear penalty function and local trust region). MINOS usually requires less of its iterations than does XS, but evidently works harder on each.

Pivots (minor iterations) for MINOS represent classical basis exchanges (a superbasic variable becomes basic, replacing a basic variable which becomes nonbasic). XS pivots are counted in several varieties, including logical cases in which no basis change or objective function improvement takes place but the logical composition of the solution changes. XS

usually requires more of its pivots than does MINOS, but apparently executes each of them faster.

Iteration counts do not appear to be adeuate general measures of efficiency for dissimilar optimization algorithms, or even alternate implementations of the same algorithm.

E. NUMBER OF FUNCTION EVALUATIONS

For the objective vunction precision specified in these tests, MINOS has generally required less function (/gradient) calls than XS. This difference can be crucial for problems in which function generation requires the vast majority of computing effort. However, a paradox is apparent in that such problems often have no closed-firm derivatives: desirable for (quasi-Newton) second-order algorithms.[6] In most cases, XS provides 2-3 decimal place precision in the objective function value with less function evaluations than MINOS, but MINOS produces 3-5 place precision with less function calls than XS. MINOS usually requires more function evaluations than XS to yield a feasible solution.

For most classes of functions (polynomial, exponential, etc.), the gradient is of the same class as the function; the function calls (and therefore the required CPU time) to determine gradient information are approximately the same for both analytic and numerical methods.

---

[6] A conversation late in this research with Professor G. Vanderplatz, Dept. of Mechanical Engineering, Naval Postgraduage School, revealed that his widely used optimization codes for engineering design problems have been evaluated using the number of function calls as the exclusive gauge of efficiency, that closed-form gradients are rarely available for such problems, and that empirical engineering data resolution may often be as poor as $\pm$ 20 percent.

F. USER FRIENDLINESS

Because of its documentation and intended use, it was expected from the start of the study that MINOS would be the system most amenable to the user and the results bear this out.

1. Ease of Setup

This was one area in which MINOS was not necessarily better. The requirements for providing analytic gradients proved to be very time consuming, notwithstanding their aforementioned use as a trouble-shooting aid. In addition, the need to refer to a number of documents for constructing the SPECS and MPS files (until "templates" were constructed) was a handicap to the new user. Once the programming aids were in place, construction of a new problem became quite straightforward with a minimum of outside reference required.

Although relatively little was required of the user for initial setup on XS, becoming familiar with the techniques of using "pointer" variables required time; especially with the dearth of documentation.

Interestingly, as the user becomes more experienced with both systems, MINOS becomes easier to use with the exception of the gradient functions, while in some ways XS becomes more difficult as the profound tuning capability of the system becomes apparent and places more demands on the talents of the user/analyst (i.e., appropriate problem formulation).

2. Debug Output

Both systems are capable of producing voluminous debug listings to assist the user, and the only limitation is the knowledge and patience of the user in their interpretation.

3. Failure Mode

Both codes give comprehensive output to indicate why they fail. As long as the user is sophisticated enough in system use to understand the diagnostics, he can usually intuit the cause.

4. Robustness

Default values exist for all parameters in the SPECS file for MINOS and experience has shown these values to be very robust with little tuning required other than specifying the problem-specific size parameters. Although the explicit statement of the objective function gradient is recommended for MINOS [Ref 6: p. 18], experimentation with the objective function differencing option has revealed no significant change in solution values or CPU time between the explicit gradient and the numerical differencing representation.

Many of the XS tuning parameters are dependent on the scaling of the problem, and their robustness is in direct proportion with the user's ability to provide a well-scaled problem. This just requires reasonable care in the original problem formulation, but for complex test problems presented in completed form, reformulation and scaling can be vexing.

G. SUMMARY

After 15 months of intensive use of both codes, it is apparent that both systems have achieved what their designers intended. MINOS is a well-documented, easy-to-use code that reliably achieves excellent results on the general (NLP) problem while demanding only moderate skill of the user. Its default parameters are robust and require minimal tuning to achieve satisfactory results. Although its input files can be somewhat cumbersome to manage, they are straightforward and unlikely to

cause a confidence crisis for the inexperienced user. Constraint gradients, on the other hand, can be arduous to prepare and debug, even for simple test problems. MINOS does not have some of the more sophisticated file editing and solution options, but its developers candidly admit that this was not a design goal. MINOS is not capable of handling integer variables, but again, was not intended to do so. MINOS appears to be a significant improvement over other contemporary codes, and a most useful tool for the modeller/analyst.

XS, is also exactly what it is claimed to be: an experimental testbed for state-of-the-art optimization research. It is very easy to use, but is not intended primarily for the casual, inexperienced user. Nor is XS specifically designed for nonlinear models. XS provides many flexible file editing, problem representation, solution, and report options, but is designed for efficient custom applications to particular classes of models. The default nonlinear feature of XS, operating with one of the problem representation "templates," provides a quick-turnaround modelling enviornment.

XS is designed to accommodate problems with no analytic gradients, poor data resolution, approximated functions, and all the attendant difficulties of real-life optimization at large-scale. Problems should be formulated for XS with realistic range intervals and meaningful penalties associated with constraints. Accordingly, XS can encounter difficulties with artificial test problems presented in strictest equality form with default (rather than modelled) penalties.

The large-scale nonlinear integer capability of XS, combined with basis factorization (e.g., GUB), decomposition facilities, etc., make it

truly unique in the field of large-scale optimization. It is capable of solving problems that no other system in the world known to the author can attempt. By achieving this, XS has more than fulfilled the goals of its developers.

APPENDIX A

TEST PROBLEMS AND RESULTS

Problem 1 (Himmelblau 6)

Source: [Ref. 30]

No. of variables: 45

No. of constraints: 16 linear equality constraints

Objective function:

$$\text{Minimize: } f(x) = \sum_{k=1}^{7} \left[\sum_{j=1}^{n_k} x_{jk} \left(c_{jk} + \ln \frac{x_{jk}}{\sum_{j=1}^{n_k} x_{jk}} \right) \right]$$

Constraints:

$$g_i(x) = \sum_{k=1}^{7} \left(\sum_{j=1}^{n_k} E_{ijk} x_{ijk} \right) - b_i = 0 \quad i = 1,\ldots,16$$

$$x_{jk} \geq 0 \quad j = 1,\ldots,n_k \quad k = 1,\ldots,7$$

b_j's and c_{jk}'s for Problem 1

i	b_i	j	k	c_{jk}	j	k	c_{jk}
1	0.6529581	1	1	0.0	6	3	0.0
2	0.281941	2	1	-7.69	7	3	2.2435
3	3.705233	3	1	-11.52	8	3	0.0
4	47.00022	4	1	-36.60	9	3	-39.39
5	47.02972	1	2	-10.94	10	3	-21.49
6	0.08005	2	2	0.0	11	3	-32.84
7	0.08813	3	2	0.0	12	3	6.12
8	0.04829	4	2	0.0	13	3	0.0
9	0.0155	5	2	0.0	14	3	0.0
10	0.0211275	6	2	0.0	15	3	-1.9028
11	0.0022725	7	2	0.0	16	3	-2.8889
12	0.0	8	2	2.5966	17	3	-3.3622
13	0.0	9	2	-39.39	18	3	-7.4854
14	0.0	10	2	-21.35	1	4	-15.639
15	0.0	11	2	-32.84	2	4	0.0
16	0.0	12	2	6.26	3	4	21.81
		13	2	0.0	1	5	-16.79
		1	3	10.45	2	5	0.0
		2	3	0.0	3	5	18.9779
		3	3	-0.50	1	6	0.0
		4	3	0.0	2	6	11.959
		5	3	0.0	1	7	0.0
					2	7	12.899

E_{ijk} Data for Problem 1

x^i_{jk}	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
x_{11}	1															
x_{21}		1														
x_{31}			1													
x_{41}				1	1											
x_{12}	1															
x_{22}		1														
x_{32}			1													
x_{42}				1								1				
x_{52}					1							-1				
x_{62}						1						-1				
x_{72}							1					1				
x_{82}								1				1				
x_{92}				1	1											
$x_{10,2}$		1			1							-1				
$x_{11,2}$		1		1	1											
$x_{12,2}$		1		-1	1							-2				
$x_{13,2}$									1			-1				
x_{13}	1															
x_{23}		1														
x_{33}			1													
x_{43}				1												
x_{53}					1											
x_{63}						1										
x_{73}							1									
x_{83}								1								
x_{93}				1	1											
$x_{10,3}$		1			1											
$x_{11,3}$		1		1	1											
$x_{12,3}$		1		-1	1											
$x_{13,3}$										1						

x^i_{jk}	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$x_{14,3}$											1	-4				
$x_{15,3}$	1										1	-3	-1			
$x_{16,3}$	2										1	-2	-2			
$x_{17,3}$	3										1	-1	-3			
$x_{18,3}$	4										1		-4			
x_{14}				1									1			
x_{24}													1			
x_{34}				-1									1		-4	
x_{15}				1										1		
x_{25}														1		
x_{35}				-1										1		-4
x_{16}															1	
x_{26}		1		-1											1	
x_{17}																1
x_{27}		1		-1												1

Alternate Initial Points:

a) $x_{jk} = 0.01 \qquad j=1,\ldots,n_k \qquad k=1,\ldots,7$

b) $x_{jk} = 1.0 \qquad j=1,\ldots,n_k \qquad k=1,\ldots,7$

Results for Problem 1

	Initial	MINOS	XS
f(x)	-30.958	-1895.4	-1910.0
x_{11}	0.1	0.64387	0.0
x_{21}	0.1	0.0	0.25803
x_{31}	0.1	3.7052	3.7052
x_{41}	0.1	0.0	0.32129
x_{12}	0.1	0.0	0.64736
x_{22}	0.1	0.0	0.0
x_{32}	0.1	0.0	0.0
x_{42}	0.1	0.0	0.0
x_{52}	0.1	0.72630D-01	0.0
x_{62}	0.1	0.0	0.17279D-01
x_{72}	0.1	0.88130D-01	0.36270D-01
x_{82}	0.1	0.0	0.0
x_{92}	0.1	46.675	33.231
$x_{10,2}$	0.1	0.0	0.0
$x_{11,2}$	0.1	0.0	0.0
$x_{12,2}$	0.1	0.0	0.0
$x_{13,2}$	0.1	0.15500D-01	0.15500D-01
x_{13}	0.1	0.0	0.0
x_{23}	0.1	0.0	0.0
x_{33}	0.1	0.0	0.0
x_{43}	0.1	0.0	0.0
x_{53}	0.1	0.0	0.0
x_{63}	0.1	0.80050D-01	0.62771D-01
x_{73}	0.1	0.0	0.51860D-01
x_{83}	0.1	0.48290D-01	0.48290D-02
x_{93}	0.1	0.30577	13.476
$x_{10,3}$	0.1	0.0	0.15170D-02
$x_{11,3}$	0.1	0.0	0.0
$x_{12,3}$	0.1	0.28194	0.0
$x_{13,3}$	0.1	0.21127D-01	0.21127D-01
$x_{14,3}$	0.1	0.0	0.11682D-03

	Initial	MINOS	XS
$x_{15,3}$	0.1	0.0	0.10080D-02
$x_{16,3}$	0.1	0.0	0.0
$x_{17,3}$	0.1	0.0	0.0
$x_{18,3}$	0.1	0.22725D-02	0.11477D-02
x_{14}	0.1	0.0	0.0
x_{24}	0.1	0.0	0.0
x_{34}	0.1	0.90900D-02	0.55987D-02
x_{15}	0.1	0.0	0.0
x_{25}	0.1	0.0	0.0
x_{35}	0.1	0.0	0.0
x_{16}	0.1	0.3636D-01	0.0
x_{26}	0.1	0.0	0.22395D-01
x_{17}	0.1	0.0	0.0
x_{27}	0.1	0.0	0.0
$g_1(x)$	0.647	0.65296	0.65296
$g_2(x)$	0.818	0.28194	0.28194
$g_3(x)$	-3.405	3.7052	3.7052
$g_4(x)$	-46.70	47.000	47.000
$g_5(x)$	-45.93	47.030	47.030
$g_6(x)$	0.12	0.80050D-01	0.80050D-01
$g_7(x)$	0.112	0.88130D-01	0.88130D-01
$g_8(x)$	0.152	0.48290D-01	0.48290D-01
$g_9(x)$	0.085	0.15500D-01	0.15500D-01
$g_{10}(x)$	0.079	0.21127D-01	0.21127D-01
$g_{11}(x)$	0.498	0.22725D-02	0.22725D-02
$g_{12}(x)$	-1.3	0.0	0.0
$g_{13}(x)$	-0.7	0.0	0.0
$g_{14}(x)$	0.3	0.0	0.0
$g_{15}(x)$	-0.2	0.0	0.0
$g_{16}(x)$	-0.2	0.0	0.0

Problem 2 (Himmelblau 4A)

Source: [Ref. 31, 32]

No. of independent variables: 10

No. of constraints: 3 nonlinear equality constraints

Objective function:

$$\text{Minimize: } f(x) = \sum_{i=1}^{10} \left[e^{x_i} \left(c_i + x_i - \ln \sum_{i=1}^{10} e^{x_i} \right) \right]$$

Constraints:

$$g_1(x) = e^{x_1} + 2e^{x_2} + 2e^{x_3} + e^{x_6} + e^{x_{10}} - 2 = 0$$

$$g_2(x) = e^{x_4} + 2e^{x_5} + e^{x_6} + e^{x_7} - 1 = 0$$

$$g_3(x) = e^{x_3} + e^{x_7} + e^{x_8} + 2e^{x_9} + e^{x_{10}} - 1 = 0$$

where $c_1 = -6.089$ $c_2 = -17.164$ $c_3 = -34.054$ $c_4 = -5.914$

$c_5 = -24.721$ $c_6 = -14.986$ $c_7 = -24.1000$

$c_8 = -10.708$ $c_9 = -26.662$ $c_{10} = -22.179$

Results for Problem 2

	Initial	MINOS	XS
$f(x)$	-21.015	-47.761	-48.113
x_1	-2.3	-3.2023	-3.3543
x_2	-2.3	-1.9124	-1.9456
x_3	-2.3	-0.24443	-0.22381
x_4	-2.3	-6.5612	-7.5599
x_5	-2.3	0.72310	-0.72055
x_6	-2.3	-7.2742	-7.1958
x_7	-2.3	-3.5973	-3.5789
x_8	-2.3	-4.0203	-5.1213
x_9	-2.3	-3.2884	-3.0806
x_{10}	-2.3	-2.3344	-2.4085
$g_1(x)$	-1.298	0.0	0.10383D-01
$g_2(x)$	-0.499	0.0	0.21430D-02
$g_3(x)$	-0.398	0.0	0.15153D-01

Alternate Initial Points:

a) $x_i = 2.0$ $i=1,\ldots,10$

b) $x_i = -5.0$ $i=1,\ldots,10$

Problem 3 (Himmelblau 18)

Source: [Ref. 33]

No. of variables: 15

No. of constraints: 5 nonlinear inequality constraints

15 bounds on independent variables

Objective function:

$$\text{Maximize: } f(x) = \sum_{i=1}^{10} b_i x_i - \sum_{j=1}^{5} \sum_{i=1}^{5} yz - 2 \sum_{j=1}^{5} d_j z^3$$

where $y = c_{ij} x_{(10+i)}$

and $z = x_{(10+i)}$

Constraints:

$$2 \sum_{i=1}^{2} y + 3d_j z^2 + e_j - \sum_{i=1}^{10} a_{ij} x_i \geq 0 \qquad j = 1,\ldots,5$$

$$x_i \geq 0 \qquad i = 1,\ldots,15$$

Data for Problem 3

j	1	2	3	4	5
e_j	-15	-27	-36	-18	-12
c_{1j}	30	-20	-10	32	-10
c_{2j}	-20	39	-6	-31	32
c_{3j}	-10	-6	10	-6	-10
c_{4j}	32	-31	-6	39	-20
c_{5j}	-10	32	-10	-20	30
d_j	4	8	10	6	2
a_{1j}	-16	2	0	1	0
a_{2j}	0	-2	0	0.4	2
a_{3j}	-3.5	0	2	0	0
a_{4j}	0	-2	0	-4	-1
a_{5j}	0	-9	-2	1	-2.8
a_{6j}	2	0	-4	0	0
a_{7j}	-1	-1	-1	-1	-1
a_{8j}	-1	-2	-3	-2	-1
a_{9j}	1	2	3	4	5
a_{10j}	1	1	1	1	1

b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
-40	-2	-.25	-4	-4	-1	-40	-60	5	1

Alternate Initial Points:

a) $x_i = 5.0$ $\quad i=1,\ldots,15$

b) $x_i = 15.0$ $\quad i=1,\ldots,5$

Results for Problem 3

	Initial	MINOS	XS
$f(x)$	2400.1	-32.349	-34.228
x_1	1.0D-04	0.0	0.0
x_2	1.0D-04	0.0	0.0
x_3	1.0D-04	5.1740	6.3781
x_4	1.0D-04	0.0	0.19670
x_5	1.0D-04	3.0611	3.6463
x_6	1.0D-04	11.840	12.886
x_7	6.0D-01	0.0	0.0
x_8	1.0D-04	0.0	0.0
x_9	1.0D-04	0.10390	0.0
x_{10}	1.0D-04	0.0	0.0
x_{11}	1.0D-04	0.30000	0.20177D-01
x_{12}	1.0D-04	0.33347	0.0
x_{13}	1.0D-04	0.40000	0.0
x_{14}	1.0D-04	0.42831	0.36088
x_{15}	1.0D-04	0.22396	0.26740
$g_1(x)$	4.5D-01	0.0	0.51426
$g_2(x)$	3.3D-01	0.0	0.14235
$g_3(x)$	2.4D-01	0.0	0.0
$g_4(x)$	4.2D-01	0.0	0.22845
$g_5(x)$	4.8D-01	0.0	0.40583D-01

Problem 4 (Himmelblau 16)

Source: [Ref. 34]

No. of variables: 9

No. of constraints: 13 nonlinear inequality constraints

1 upper bound

Objective function:

maximize:

$$f(x) = 0.5(x_1x_4 - x_2x_3 + x_3x_9 - x_5x_9 + x_5x_8 - x_6x_7)$$

Constraints:

$$1 - x_3^2 - x_4^2 \geq 0$$

$$1 - x_9^2 \geq 0$$

$$1 - x_5^2 - x_6^2 \geq 0$$

$$1 - x_1^2 - (x_2 - x_9)^2 \geq 0$$

$$1 - (x_1 - x_5)^2 - (x_2 - x_6)^2 \geq 0$$

$$1 - (x_1 - x_7)^2 - (x_2 - x_8)^2 \geq 0$$

$$1 - (x_3 - x_5)^2 - (x_4 - x_6)^2 \geq 0$$

$$1 - (x_3 - x_7)^2 - (x_4 - x_8)^2 \geq 0$$

$$1 - x_7^2 - (x_8 - x_9)^2 \geq 0$$

$$x_1x_4 - x_2x_3 \geq 0$$

$$x_3x_9 \geq 0$$

$$-x_5x_9 \geq 0$$

$$x_5x_8 - x_6x_7 \geq 0$$

$$x_9 \geq 0$$

Results for Problem 4

	Initial	MINOS	XS
$f(x)$	0.0	-0.86603	-0.86581
x_1	1.0	-0.78028D-01	0.93171
x_2	1.0	-0.67479	0.36276
x_3	1.0	0.82437	0.16281
x_4	1.0	-0.56605	0.98669
x_5	1.0	-0.78028D-01	0.93573
x_6	1.0	-0.99695	0.35193
x_7	1.0	0.82437	0.15144
x_8	1.0	-0.24388	0.98818
x_9	1.0	0.32216	0.0
$g_1(x)$	-1.0	0.0	-0.73055D-04
$g_2(x)$	0.0	0.89621	1.0000
$g_3(x)$	-1.0	0.0	0.55734D-03
$g_4(x)$	0.0	0.0	0.32201D-03
$g_5(x)$	1.0	0.89621	0.99987
$g_6(x)$	1.0	0.0	0.31116D-04
$g_7(x)$	1.0	0.0	-0.31858D-03
$g_8(x)$	1.0	0.89621	0.99987
$g_9(x)$	0.0	0.0	0.57312D-03
$g_{10}(x)$	0.0	0.60044	0.86025
$g_{11}(x)$	1.0	0.26558	0.0
$g_{12}(x)$	-1.0	0.25138D-01	0.0
$g_{13}(x)$	0.0	0.84089	0.87137

Alternative Initial Points:

a) $x_i = 1.0$ $i=1,\ldots,8$ $x_9 = 0.0$

b) $x_i = 5.0$ $i=1,\ldots,9$

Problem 5 (Himmelblau 20)

Source: [Ref. 35]

No. of variables: 24

No. of constraints: 12 nonlinear equality constraints

2 linear equality constraints

6 nonlinear inequality constraints

24 bounds on independent variables

Objective function:

$$\text{minimize:} \quad f(x) = \sum_{i=1}^{24} a_i x_i$$

Constraints:

$$g_i(x) = \frac{x_{(i+12)}}{b_{(i+12)} \sum_{j=13}^{24} x_j / b_j} - \frac{c_i x_i}{40 b_i \sum_{j=1}^{12} x_j / b_j} = 0 \quad i = 1,\ldots,12$$

$$g_{13}(x) = \sum_{i=1}^{24} x_i - 1 = 0$$

$$g_{14}(x) = \sum_{i=1}^{12} \frac{x_i}{d_i} + f \sum_{i=13}^{24} \frac{x_i}{b_i} - 1.671 = 0$$

where $f = (0.7302)\ (530)\ \frac{14.7}{40}$

$$g_{(i+14)}(x) = \frac{-x_i + x_{(i+12)}}{\sum_{j=1}^{24} x_j} + e_i \geq 0 \qquad i = 1,2,3$$

$$g_{(i+14)}(x) = \frac{-x_{(i+3)} + x_{(i+15)}}{\sum_{j=1}^{24} x_j} + e_i \geq 0 \qquad i = 4,5,6$$

$$x_i \geq 0 \qquad i = 1,\ldots,24$$

Data for Problem 5

i	a_i	b_i	c_i	d_i	e_i
1	0.0693	44.094	123.7	31.244	0.1
2	0.0577	58.12	31.7	36.12	0.3
3	0.05	58.12	45.7	34.784	0.4
4	0.20	137.4	14.7	92.7	0.3
5	0.26	120.9	84.7	82.7	0.6
6	0.55	170.9	27.7	91.6	0.3
7	0.06	62.501	49.7	56.708	
8	0.10	84.94	7.1	82.7	
9	0.12	133.425	2.1	80.8	
10	0.18	82.507	17.7	64.517	
11	0.10	46.07	0.85	49.4	
12	0.09	60.097	0.64	49.1	
13	0.0693	44.094			
14	0.0577	58.12			
15	0.05	58.12			
16	0.20	137.4			
17	0.26	120.9			
18	0.55	170.9			
19	0.06	62.501			
20	0.10	84.94			
21	0.12	133.425			
22	0.18	82.507			
23	0.10	46.07			
24	0.09	60.097			

Results for Problem 5

	Initial	MINOS	XS
$f(x)$	0.14696	0.55658D-01	0.55658D-01
x_1	0.04	0.0	0.27513D-08
x_2	0.04	0.10725	0.10725
x_3	0.04	0.11139	0.11139
x_4	0.04	0.40506D-14	0.24201D-09
x_5	0.04	0.11591D-13	0.28816D-08
x_6	0.04	0.22064D-14	0,13956D-10
x_7	0.04	0.75541D-01	0.75541D-01
x_8	0.04	0.0	0.15094D-09
x_9	0.04	0.0	0.80735D-08
x_{10}	0.04	-0.17419D-13	0.59860D-09
x_{11}	0.04	0.0	0.17049D-07
x_{12}	0.04	0.11195D-01	0.11195D-01
x_{13}	0.04	-0.10753D-16	0.55220D-07
x_{14}	0.04	0.19275	0.19275
x_{15}	0.04	0.28861	0.28861
x_{16}	0.04	0.0	0.62272D-09
x_{17}	0.04	0.0	0.39282D-07
x_{18}	0.04	0.0	0.23459D-10
x_{19}	0.04	0.21286	0.21286
x_{20}	0.04	0.94396D-20	0.14615D-09
x_{21}	0.04	0.14441D-14	0.29088D-08
x_{22}	0.04	0.0	0.14346D-08
x_{23}	0.04	0.45547D-15	0.38028D-08
x_{24}	0.04	0.40622D-03	0.40622D-03
$g_1(x)$	-2.9D-01	0.0	0.69666D-07
$g_2(x)$	2.2D-02	0.0	-0.12910D-07

$g_3(x)$	-1.5D-02	0.0	-0.81246D-06
$g_4(x)$	2.8D-02	0.0	0.26202D-09
$g_5(x)$	-5.6D-02	0.0	0.17996D-07
$g_6(x)$	1.1D-02	0.0	0.0
$g_7(x)$	-2.4D-02	0.0	0.82383D-06
$g_8(x)$	5.9D-02	0.0	0.85965D-10
$g_9(x)$	4.3D-02	0.0	0.12482D-08
$g_{10}(x)$	4.1D-02	0.0	0.86423D-09
$g_{11}(x)$	1.3D-01	0.0	0.55334D-08
$g_{12}(x)$	1.0D-02	0.0	-0.14175D-07
$g_{13}(x)$	-4.0D-02	0.0	0.0
$g_{14}(x)$	-7.3D-01	0.0	0.0
$g_{15}(x)$	1.6D-02	0.10000	0.10000
$g_{16}(x)$	2.2D-01	0.0	0.64365D-06
$g_{17}(x)$	3.2D-01	0.0	0.74807D-06
$g_{18}(x)$	2.2D-01	0.11601D-01	0.11600D-01
$g_{19}(x)$	5.2D-01	0.60000	0.60000
$g_{20}(x)$	2.2D-01	0.30000	0.30000

Alternative Initial Points:

a) $x_i = 0.08$ $i=1,\ldots,24$

b) $x_i = 0.02$ $i=1,\ldots,24$

Problem 6 and Problem 12 (Himmelblau 23)

Source: [Ref. 32, 36]

No. of independent variables: 100

No. of constraints: 12 linear constraints

100 lower bounds on the variables

Objective function:

$$\text{minimize:} \quad f(x) = \sum_{j=1}^{20} u_j \left(\prod_{i=1}^{5} a_{ij}^{x_{ij}} - 1 \right)$$

Constraints:

$$\sum_{i=1}^{5} x_{ij} \geq b_j \qquad j = 1,6,10,14,15,16,20$$

$$- \sum_{j=1}^{20} x_{ij} \geq -c_i \qquad i = 1\ldots,5$$

$$x_{ij} \geq 0 \qquad i = 1,\ldots,5 \qquad j = 1,\ldots,20$$

$x_{ij} \in$ [integer] --- Problem 12 only

Data from Problem 6

	a_{ij}'s					b_j's	u_j's
i^j	1	2	3	4	5		
1	1	.84	.96	1	.92	30	60
2	.95	.83	.95	1	.94		50
3	1	.85	.96	1	.92		50
4	1	.84	.96	1	.95		75
5	1	.85	.96	1	.95		40
6	.85	.81	.90	1	.98	100	60
7	.90	.81	.92	1	.98		35
8	.85	.82	.91	1	1		30
9	.80	.80	.92	1	1		25
10	1	.86	.95	.96	.90	40	150
11	1	1	.99	.91	.95		30
12	1	.98	.98	.92	.96		45
13	1	1	.99	.91	.91		125
14	1	.88	.98	.92	.98	50	200
15	1	.87	.97	.98	.99	70	200
16	1	.88	.98	.93	.99	35	130
17	1	.85	.95	1	1		100
18	.95	.84	.92	1	1		100
19	1	.85	.93	1	1		100
20	1	.85	.92	1	1	10	150
c_i	200	100	300	150	250		

Results for Problem 6

	Initial	MINOS	XS	XS Integer
$x(\cdot)$	100.0	----	(see next page)	----
$f(x)$	-1755.0	-1735.6	-1734.5	-1734.5
$g_1(x)$	-1770.0	50.815	50.236	50.0
$g_2(x)$	-1800.0	100.00	100.0	100.0
$g_3(x)$	-1660.0	51.132	51.133	51.0
$g_4(x)$	-1800.0	58.824	52.928	52.0
$g_5(x)$	-1680.0	70.000	70.0	70.0
$g_6(x)$	505.0	41.281	48.139	53.0
$g_7(x)$	410.0	62.414	60.401	61.0
$g_8(x)$	260.0	-200.00	-200.0	-200.0
$g_9(x)$	350.0	-100.00	-100.0	-100.0
$g_{10}(x)$	130.0	-300.00	-300.0	-300.0
$g_{11}(x)$	315.0	-150.00	-150.0	-150.0
$g_{12}(x)$	240.0	-250.00	-250.0	-250.0

Alternative Initial Points:

a) $x_{1j} = 10.0$ $x_{2j} = 5.0$ $x_{3j} = 15.0$

$x_{4j} = 7.5$ $x_{5j} = 12.5$ $j = 1,\ldots,20$

b) $x_{ij} = 10$ $i = 1,\ldots,5$ $j = 1,\ldots,20$

Weapon Type i

Target j	1	2	3	4	5	Total
1					-50.8- (50.2) 50.0	-50.8- (50.2) 50.0
2	-13.5- (33.3) 40.0	-1.4- (8.5) 7.0	5.0		-45.4- (7.9) 2.0	-60.3- (49.7) 54.0)
3					-48.6- (48.2) 48.0	-48.6- (48.2) 48.0
4		-23.5- (17.7) 20.0	1.0		(15.9) 11.0	-23.5- (33.6) 32.0
5		-20.9- (13.3) 12.0	16.0		(22.6) 16.0	-20.9- (35.9) 44.0
6	-100.0- (66.3) 76.0		(33.7) 24.0			-100.0- (100.0) 100.0
7	-39.1- (38.7) 38.0					-39.1- (38.7) 38.0
8	-27.1- (27.0) 26.0					-27.1- (27.0) 26.0
9	-20.3- (20.3) 20.0					-20.3- (20.3) 20.0
10					-51.1- (51.1) 51.0	-51.1- (51.1) 51.0

- - denotes MINOS variables

( ) denotes XS variables

No brackets denotes XS integer variables for Problem 12.

Target j	1	2	3	4	5	Total
11				-33.2- (32.2) 22.0	(1.5) 18.0	-33.2- (33.7) 40.0
12				-40.9- (40.5) 40.0		-40.9- (40.5) 40.0
13					-54.0- (52.4) 54.0	-54.0- (52.4) 54.0
14		(9.6) 10.0		-58.8- (43.3) 42.0		-58.8- (52.9) 42.0
15		-26.2- (26.5) 26.0	-43.8- (43.5) 44.0			-70.0- (70.0) 44.0
16		-24.2- (14.1) 7.0		-17.0- (34.0) 46.0		-41.2- (48.1) 46.0
17		-3.8- (10.1) 18.0	-72.0- (52.1) 27.0			-75.8- (62.2) 27.0
18	(14.4)		-57.6- (46.0) 58.0			-57.6- (60.4) 58.0
19			-64.2- (63.3) 64.0			-64.2- (64.3) 64.0
20			-62.4- (60.4) 61.0			-62.4- (60.4) 61.0
Totals	-200- (200) 200	-100- (100) 100	-300- (300) 300	-150- (150) 150	-250- (250) 250	

- - denotes MINOS variables

( ) denotes XS variables

No brackets denotes XS integer variables for Problem 12.

Problem 7 (Waterman 7)

Source: [Ref. 27]

No. of variables: 100

No. of constraints: 1 linear constraint

1 nonlinear constraint

50 lower bounds on the variables

Objective function:

$$\text{minimize:} \quad f(x) = \sum_{i=1}^{50} \frac{B_i(x_i + 50)}{x_i}$$

where

$$B_i(x_i + 50) = \frac{1}{2}\left(s_i^2 + d_i^2\right)\Phi\left[d_i/s_i\right] - \frac{s_i d_i}{2}\phi\left[d_i/s_i\right]$$

and

$$d_i = x_{(i+50)} - m_i$$

$$\phi(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$$

$$\Phi(r) = \int_r^x \phi(x)dx$$

Constraints:

$$\sum_{i=1}^{50} c_i \frac{x_i}{2} + x_{(i+50)} - m_i \leq K_1$$

$$\sum_{i=1}^{50} \frac{L_i}{x_i} \leq K_2$$

$x_i \geq 0 \quad i = 1,\ldots,50$

$K_1 = 200,000$

$K_2 = 300$

Data from Problem 7

	L_i's	c_i's	m_i's	s_i's
1	1000	1	100	100
2	1500	10	200	100
3	2000	20	300	200
4	1100	17	200	100
5	1900	23	100	100
6	700	8	200	200
7	400	12	200	200
8	1200	19	300	100
9	2000	2	500	200
10	1300	5	300	100
11	1900	21	100	100
12	900	16	200	200
13	1400	13	400	200
14	1500	19	500	300
15	2200	7	400	100
16	1700	4	300	100
17	1800	12	200	200
18	800	5	100	100
19	700	18	100	100
20	1100	16	100	100
21	1000	14	200	100
22	1800	21	200	200
23	1500	6	400	300
24	2100	6	500	100
25	1600	14	100	100

	L_i's	c_i's	m_i's	s_i's
26	700	2	100	100
27	2000	12	200	200
28	1800	3	500	300
29	1700	1	200	200
30	700	18	300	200
31	1200	19	100	100
32	1100	12	100	100
33	1700	9	500	100
34	600	8	300	100
35	400	1	200	100
36	1000	3	100	100
37	1900	17	400	300
38	1500	15	200	200
39	1400	18	400	300
40	1200	16	500	300
41	1300	5	100	100
42	1900	12	200	100
43	2000	15	300	200
44	2200	20	400	200
45	800	23	100	100
46	1900	17	200	200
47	2100	16	500	200
48	2000	4	500	300
49	500	8	100	100
50	900	12	100	100

Results for Problem 7

	Initial	MINOS	XS
f(x)	2008.2	80.744	86.215
x_1	300.0	440.22	353.95
x_2	300.0	202.72	183.778
x_3	300.0	235.98	221.42
x_4	300.0	153.51	143.77
x_5	300.0	169.57	151.49
x_6	300.0	213.66	201.95
x_7	300.0	180.91	259.13
x_8	300.0	153.31	224.45
x_9	300.0	479.48	464.12
x_{10}	300.0	248.16	259.13
x_{11}	300.0	174.01	157.05
x_{12}	300.0	203.71	210.64
x_{13}	300.0	232.75	222.45
x_{14}	300.0	286.23	464.12
x_{15}	300.0	270.84	237.75
x_{16}	300.0	304.74	272.13
x_{17}	300.0	253.87	259.13
x_{18}	300.0	203.51	182.61
x_{19}	300.0	131.70	224.16
x_{20}	300.0	156.08	142.31
x_{21}	300.0	156.98	143.74
x_{22}	300.0	228.41	220.12
x_{23}	300.0	336.01	464.12
x_{24}	300.0	282.50	252.25
x_{25}	300.0	185.38	166.27
x_{26}	300.0	276.92	246.86
x_{27}	300.0	261.94	259.13

	Initial	MINOS	XS
x_{28}	300.0	432.01	602.37
x_{29}	300.0	500.00	462.028
x_{30}	300.0	192.82	187.38
x_{31}	300.0	153.31	139.08
x_{32}	300.0	169.95	153.28
x_{33}	300.0	221.19	211.62
x_{34}	300.0	155.56	146.33
x_{35}	300.0	290.95	258.89
x_{36}	300.0	272.94	242.92
x_{37}	300.0	300.04	464.12
x_{38}	300.0	230.70	220.40
x_{39}	300.0	283.96	464.12
x_{40}	300.0	279.00	464.12
x_{41}	300.0	248.16	222.23
x_{42}	300.0	208.39	186.85
x_{43}	300.0	249.02	233.13
x_{44}	300.0	241.68	228.57
x_{45}	300.0	130.00	118.90
x_{46}	300.0	239.61	242.71
x_{47}	300.0	249.04	464.12
x_{48}	300.0	409.47	602.37
x_{49}	300.0	146.09	132.06
x_{50}	300.0	158.39	144.48
x_{51}	300.0	297.80	304.94
x_{52}	300.0	334.36	337.09
x_{53}	300.0	554.47	556.60
x_{54}	300.0	322.41	323.76
x_{55}	300.0	202.27	208.06
x_{56}	300.0	543.68	544.60

	Initial	MINOS	XS
x_{57}	300.0	523.86	488.01
x_{58}	300.0	417.05	393.27
x_{59}	300.0	889.14	887.20
x_{60}	300.0	456.24	451.26
x_{61}	300.0	205.63	210.34
x_{62}	300.0	488.59	480.47
x_{63}	300.0	695.20	694.72
x_{64}	300.0	918.47	844.03
x_{65}	300.0	537.44	543.22
x_{66}	300.0	457.00	462.07
x_{67}	300.0	494.59	486.77
x_{68}	300.0	264.64	269.03
x_{69}	300.0	227.02	227.42
x_{70}	300.0	224.53	227.08
x_{71}	300.0	330.59	332.73
x_{72}	300.0	452.93	451.28
x_{73}	300.0	944.49	898.59
x_{74}	300.0	642.53	646.69
x_{75}	300.0	222.68	227.31
x_{76}	300.0	289.05	292.90
x_{77}	300.0	491.82	486.77
x_{78}	300.0	1095.5	1051.1
x_{79}	300.0	636.80	640.93
x_{80}	300.0	582.94	581.52
x_{81}	300.0	217.05	221.75
x_{82}	300.0	234.08	237.51
x_{83}	300.0	635.20	633.51

	Initial	MINOS	XS
x_{84}	300.0	456.12	456.43
x_{85}	300.0	413.00	416.54
x_{86}	300.0	273.58	278.46
x_{87}	300.0	827.17	759.29
x_{88}	300.0	483.20	483.11
x_{89}	300.0	826.89	748.54
x_{90}	300.0	945.00	867.96
x_{91}	300.0	256.24	261.13
x_{92}	300.0	324.47	328.27
x_{93}	300.0	576.26	577.86
x_{94}	300.0	652.20	653.19
x_{95}	300.0	215.75	219.24
x_{96}	300.0	468.28	462.97
x_{97}	300.0	770.32	704.44
x_{98}	300.0	1068.8	1016.5
x_{99}	300.0	258.80	262.13
x_{100}	300.0	237.33	240.31
g_1	80650.0	0.34840D+06	0.2000D+06
g_2	300.0	300.0	300.00

Alternate Initial Points:

a) $x_i = 10 \quad i = 1,\ldots,100$

b) $x_i = 1000 \quad i = 1,\ldots,100$

Problem 8 (Waterman 8)

Source: [Ref. 28]

No. of variables: 46

No. of constraints: 1 nonlinear inequality constraint

1 linear equality constraint

46 lower bounds on the variables

Objective function:

$$\text{minimize:} \quad f(x) = \sum_{i=1}^{46} \frac{x_i}{T}\left(\ln \frac{x_i}{T}\right)$$

Constraints:

$$\sum_{i=1}^{46} x_i = T$$

$$\sum_{i=1}^{46} c_i y_i + ad_i y_i^b \leq S$$

where

$$y_i = x_i + \sum_{j \in A(i)} x_j$$

A(i) consists of all arcs (in Figure 7) that converge directly and indirectly upon node i.

$$x_i \geq 0 \qquad i = 1,\ldots,46$$

Data from Problem 8

$a = 0.05$ $T = 500$

$b = 1.50$ $S = 10000$

$c_1 = 5.0$	$c_2 = 4.0$	$c_3 = 5.0$
$c_4 = 7.0$	$c_5 = 7.0$	$c_6 = 8.0$
$c_7 = 6.0$	$c_8 = 4.0$	$c_9 = 3.0$
$c_{10} = 12.0$	$c_{11} = 14.0$	$c_{12} = 10.0$
$c_{13} = 21.0$	$c_{14} = 23.0$	$c_{15} = 8.0$
$c_{16} = 9.0$	$c_{17} = 10.0$	$c_{18} = 13.0$
$c_{19} = 20.0$	$c_{20} = 5.0$	$c_{21} = 8.0$
$c_{22} = 6.0$	$c_{23} = 3.0$	$c_{24} = 8.0$
$c_{25} = 11.0$	$c_{26} = 6.0$	$c_{27} = 18.0$
$c_{28} = 15.0$	$c_{29} = 10.0$	$c_{30} = 8.0$
$c_{31} = 8.0$	$c_{32} = 14.0$	$c_{33} = 11.0$
$c_{34} = 12.0$	$c_{35} = 14.0$	$c_{36} = 18.0$
$c_{37} = 16.0$	$c_{38} = 9.0$	$c_{39} = 2.0$
$c_{40} = 8.0$	$c_{41} = 11.0$	$c_{42} = 12.0$
$c_{43} = 11.0$	$c_{44} = 18.0$	$c_{45} = 20.0$
	$c_{46} = 13.0$	

$d_i = c_i$ $i = 1,\ldots,46$

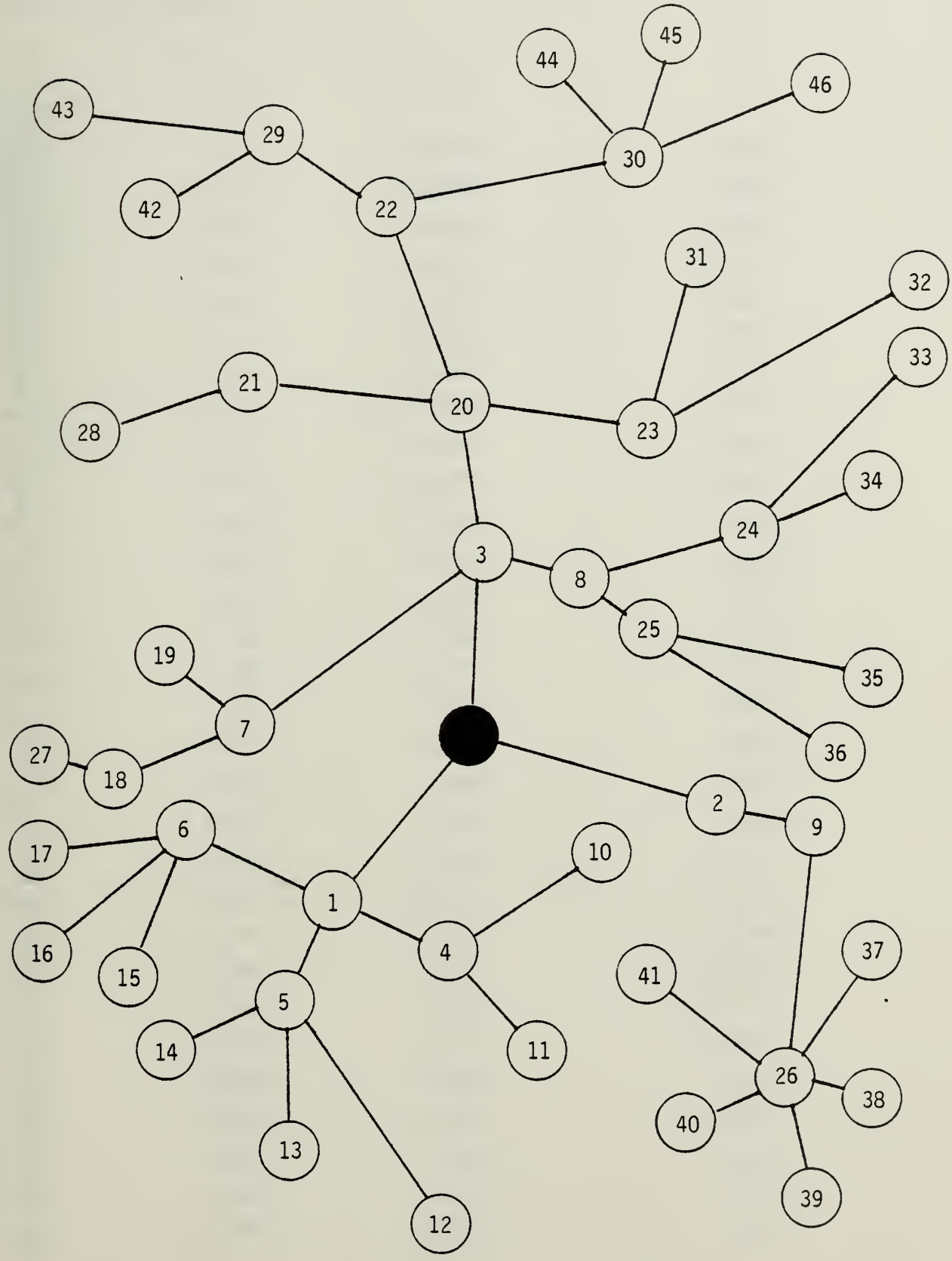


Fig. 7 Transportation Routes for Problem 8

Results for Problem 8

	Initial	MINOS	XS
f(x)	-3.249	-3.4685	-3.4519
x_1	40.000	38.690	33.766
x_2	26.667	44.648	38.106
x_3	86.667	36.171	57.241
x_4	10.000	19.468	18.032
x_5	13.333	19.357	17.223
x_6	13.333	17.140	15.654
x_7	13.333	20.084	17.869
x_8	23.333	23.498	20.499
x_9	23.333	31.411	28.061
x_{10}	3.333	7.2334	7.5175
x_{11}	3.333	6.2199	6.4508
x_{12}	3.333	8.3944	9.0758
x_{13}	3.333	3.7182	3.7996
x_{14}	3.333	3.2273	3.3423
x_{15}	3.333	8.7613	9.3672
x_{16}	3.333	8.0998	8.2227
x_{17}	3.333	7.4944	6.9622
x_{18}	6.667	6.7670	6.4252
x_{19}	3.333	4.1300	4.5276
x_{20}	46.667	20.064	25.446
x_{21}	6.667	9.9835	9.2984
x_{22}	26.667	11.205	12.408
x_{23}	10.000	15.057	13.727
x_{24}	10.000	11.270	10.197
x_{25}	10.000	8.9236	8.4804
x_{26}	20.000	16.456	15.948
x_{27}	3.333	1.7413	2.0711
x_{28}	3.333	3.1113	3.2189
x_{29}	10.000	4.8377	4.8102

	Initial	MINOS	XS
x_{30}	13.333	5.6615	5.9154
x_{31}	3.333	7.7517	7.3860
x_{32}	3.333	4.9097	4.6200
x_{33}	3.333	4.6872	4.4888
x_{34}	3.333	4.3489	4.7025
x_{35}	3.333	3.0116	3.2189
x_{36}	3.333	2.2564	2.3751
x_{37}	3.333	4.5998	4.3212
x_{38}	3.333	7.7970	7.7524
x_{39}	3.333	13.802	13.059
x_{40}	3.333	8.4329	9.0575
x_{41}	3.333	6.6818	6.2221
x_{42}	3.333	1.9479	2.0307
x_{43}	3.333	2.0952	2.0711
x_{44}	3.333	1.4707	1.4787
x_{45}	3.333	1.2772	1.4787
x_{46}	3.333	2.1051	2.0711
$g_1(x)$	-10080.33	10000.0	9999.9
$g_2(x)$	-500.00	500.00	500.0

Alternate Initial Points:

a) $x_i = 10.87$ $i = 1,\ldots,46$

b) $x_i = 85.0$ $i = 1,2,3$

$x_i = 10.0$ $i = 4,\ldots,23$

$x_i = 3.26$ $i = 24,\ldots,46$

Problem 9 (Dean 9--Game)

Source: Author

No. of independent variables: 2

No. of constraints: 1 nonlinear equality constraint

Objective function:

maximize: $x_1 \cdot x_2$

Constraints:

$$\frac{x_1^2}{(30)^2} + \frac{x_2^2}{(23)^2} = 1$$

$$x_1, x_2 \geq 0$$

Results for Problem 9

	Initial	MINOS	XS
$f(x)$	0.0	345.00	344.94
x_1	0.0	21.213	21.091
x_2	40.0	16.263	16.355
$g_1(x)$	3.0246	1.0000	0.99989

Problem 10 (Dean--Sortie)

Source: [Ref. 37]

No. of independent variables: 793

No. of constraints: 81 linear inequality constraints

Objective function:

$$\sum_{j=1}^{J} V_j(K_j - D_j)$$

where:

$$K_j = \frac{T_j}{C_j} \left\{ 1 - \exp \left[- \frac{C_j}{T_j} \left(\alpha_j + \sum_{i=1}^{I} P_{ij} X_{ij} \right) \right] \right\}$$

Constraints:

$$\sum_{j=1}^{J} x_{ij} \leq S_i, \qquad i = 1,\ldots,I$$

$$- \frac{T_j}{C_j} \log \left(1 - \frac{C_j}{T_j} l_j\right) - \alpha_j \leq \sum_{i=1}^{I} P_{ij} X_{ij} \leq - \frac{T_j}{C_j} \log (1 - C_j) - \alpha_j$$

$$j = 1,\ldots,J$$

$x_{ij} \leq S_i$, for all i and j

$$\sum_{j \epsilon J_n} X_{ij} \geq \theta_n S_i, \qquad n = 1,2,\ldots, \text{no. of side constraints}$$

$$\sum_{j \epsilon J_m} X_{ij} \leq \theta_m S_i, \qquad m = 1,2,\ldots, \text{no. of side constraints}$$

n	i	J_n	θ_n
1	1	31	0.02
2	7	32	0.02
3	7	1,6,7,11,13 16,19,20,27	0.02
4	11	23,24,25,26	0.02
5	12	23,24,25,26	0.02

m	i	J_m	θ_m
1	2	30,47,51 53,60,61	0.15
2	8	30,47,51 53,60,61	0.25

Problem Data

P_{ij}	i = 1	i = 2	i = 3	i = 4
j = 1	0.0	0.1268	0.3477	0.6705
j = 2	0.0	0.0164	0.0344	0.0676
j = 3	0.0	0.0361	0.2877	0.4399
j = 4	0.0	0.0318	0.2948	0.4482
j = 5	0.0	0.0	0.4693	0.5089
j = 6	0.0	0.0616	0.0616	0.0616
j = 7	0.0	0.0369	0.1101	0.1983
j = 8	0.0	0.0406	0.0406	0.0639
j = 9	0.0	0.0530	0.0557	0.0931
j = 10	0.0	0.4104	0.6140	0.7569
j = 11	0.0	0.0132	0.0530	0.0827
j = 12	0.0	0.3559	0.3559	0.3559
j = 13	0.0	0.0236	0.0403	0.0658
j = 14	0.0	0.0507	0.3782	0.5373
j = 15	0.0	0.0	0.1537	0.2514
j = 16	0.0	0.0510	0.1281	0.2072
j = 17	0.0	0.0988	0.1079	0.1930
j = 18	0.0	0.0157	0.0537	0.0997
j = 19	0.0	0.6347	0.6347	0.6347
j = 20	0.0	0.0984	0.1170	0.2149
j = 21	0.0	0.0024	0.0024	0.0039
j = 22	0.0	0.2226	0.7555	0.5080
j = 23	0.0	0.0044	0.0058	0.0100
j = 24	0.0	0.0052	0.0121	0.0221
j = 25	0.0	0.0052	0.0121	0.0221
j = 26	0.0	0.0034	0.0083	0.0152
j = 27	0.0	0.1404	0.2191	0.3602
j = 28	0.0	0.0877	0.0928	0.1122
j = 29	0.8082	0.0053	0.0053	0.0053
j = 30	0.2530	0.1785	0.1662	0.3307
j = 31	0.1185	0.1401	0.1401	0.1401
j = 32	0.2837	0.3033	0.3033	0.3033
j = 33	0.3310	0.786	0.1799	0.3378
j = 34	0.3409	0.0988	0.1550	0.2810
j = 35	1.3416	0.5810	1.3693	0.9057
j = 36	0.0	0.0	0.2872	0.2872
j = 37	0.0	0.0	0.0	0.0
j = 38	0.0	0.0	0.0	0.0
j = 39	0.0	0.0	0.0	0.0
j = 40	0.0	0.9123	0.9123	0.9123
j = 41	0.0	0.3888	0.3888	0.3888
j = 42	0.0	0.0	0.0	0.0
j = 43	0.0	0.0	0.0	0.0
j = 44	0.2997	0.1771	0.1898	0.2126
j = 45	0.2806	0.3491	0.3491	0.3491

P_{ij}	i = 5	i = 6	i = 7
j = 1	0.1286	0.0	0.1861
j = 2	0.1104	0.0	0.1050
j = 3	0.0361	0.0	0.0564
j = 4	0.0318	0.0	0.0507
j = 5	0.0080	0.0	0.0160
j = 6	0.0616	0.0	0.0784
j = 7	0.0369	0.0	0.0610
j = 8	0.0406	0.0	0.0507
j = 9	0.0530	0.0	0.0599
j = 10	0.4104	0.0	0.4493
j = 11	0.0132	0.0	0.0132
j = 12	0.3559	0.0	0.4527
j = 13	0.0236	0.0	0.0362
j = 14	0.0507	0.0	0.0921
j = 15	0.0030	0.0	0.0030
j = 16	0.0510	0.0	0.0759
j = 17	0.2039	0.0	0.1904
j = 18	0.0157	0.0	0.0207
j = 19	1.1287	0.6	0.8565
j = 20	0.3248	0.0	0.3184
j = 21	0.0024	0.0	0.0024
j = 22	1.4122	0.0	1.3749
j = 23	0.0044	0.0	0.0036
j = 24	0.0052	0.0	0.0052
j = 25	0.0052	0.0	0.0052
j = 26	0.0034	0.0	0.0034
j = 27	0.1404	0.0	0.1509
j = 28	0.0877	0.0	0.1056
j = 29	0.8078	0.0	0.8078
j = 30	0.2472	0.0	0.2464
j = 31	0.1401	0.0	0.1413
j = 32	0.3033	0.0	0.2591
j = 33	0.3202	0.0	0.3044
j = 34	0.3273	0.0	0.3000
j = 35	0.8938	0.0	0.7301
j = 36	0.2872	0.0	0.4931
j = 37	0.0	0.1988	0.0
j = 38	0.0	0.6822	0.0
j = 39	0.0	0.6702	0.0
j = 40	0.9123	0.0	1.0000
j = 41	0.3888	0.0	0.6217
j = 42	0.0	0.5679	0.0
j = 43	0.0	0.7527	0.0
j = 44	0.2177	0.0	0.2656
j = 45	0.3491	0.0	0.4574

P_{ij}	i = 8	i = 9	i = 10
j = 1	0.1797	0.6223	0.1818
j = 2	0.0244	0.0644	0.1036
j = 3	0.0723	0.4157	0.0727
j = 4	0.0686	0.4240	0.0690
j = 5	0.0212	0.4546	0.0222
j = 6	0.0617	0.0617	0.0629
j = 7	0.0716	0.1835	0.0733
j = 8	0.0486	0.0622	0.0511
j = 9	0.0590	0.0888	0.0613
j = 10	0.4330	0.7112	0.4467
j = 11	0.0132	0.0903	0.0132
j = 12	0.4285	0.4285	0.4385
j = 13	0.0285	0.0647	0.0296
j = 14	0.1133	0.5443	0.1160
j = 15	0.0040	0.2511	0.0100
j = 16	0.0574	0.2185	0.0585
j = 17	0.0648	0.1727	0.1795
j = 18	0.0203	0.0883	0.0209
j = 19	0.6087	0.6087	0.7022
j = 20	0.1076	0.2049	0.3107
j = 21	0.0024	0.0037	0.0024
j = 22	0.3077	1.4168	1.2910
j = 23	0.0036	0.0088	0.0036
j = 24	0.0052	0.0205	0.0052
j = 25	0.0052	0.0205	0.0052
j = 26	0.0028	0.0139	0.0028
j = 27	0.1511	0.3435	0.1570
j = 28	0.1061	0.1151	0.1099
j = 29	0.0047	0.0047	0.8096
j = 30	0.1314	0.3280	0.2557
j = 31	0.0660	0.0660	0.0660
j = 32	0.2521	0.2521	0.2521
j = 33	0.0610	0.3290	0.3067
j = 34	0.0704	0.2658	0.3091
j = 35	0.6724	0.8096	0.6919
j = 36	0.2858	0.2858	0.2953
j = 37	0.0	0.0	0.0
j = 38	0.0	0 0	0.0
j = 39	0.0	0.0	0.0
j = 40	0.8753	0.8753	0.9032
j = 41	0.4618	0.4618	0.4761
j = 42	0.0	0.0	0.0
j = 43	0.0	0.0	0.0
j = 44	0.1248	0.1763	0.2322
J = 45	0.2935	0.2935	0.2954

P_{ij}	i = 11	i = 12	i = 13
j = 1	0.5609	0.0360	0.0
j = 2	0.1145	0.0070	0.0
j = 3	0.3825	0.0070	0.0
j = 4	0.3900	0.0060	0.0
j = 5	0.4765	0.0110	0.0
j = 6	0.1220	0.0760	0.0
j = 7	0.1750	0.0260	0.0
j = 8	0.0942	0.0650	0.0
j = 9	0.1011	0.0430	0.0
j = 10	0.7068	0.1800	0.0
j = 11	0.0397	0.0080	0.0
j = 12	0.4818	0.1970	0.0
j = 13	0.0552	0.0350	0.0
j = 14	0.5091	0.0390	0.0
j = 15	0.2284	0.0	0.0
j = 16	0.1434	0.0390	0.0
j = 17	0.1706	0.1210	0.0
j = 18	0.0815	0.0110	0.0
j = 19	0.7362	0.4230	0.0
j = 20	0.3022	0.1570	0.0
j = 21	0.0030	0.0020	0.0
j = 22	1.3606	0.0920	0.0
j = 23	0.0079	0.0040	0.0
j = 24	0.0172	0.0050	0.0
j = 25	0.0172	0.0050	0.0
j = 26	0.0117	0.0040	0.0
j = 27	0.3190	0.0870	0.0
j = 28	0.2051	0.0610	0.0
j = 29	0.8081	0.0020	0.3483
j = 30	0.2593	0.0498	0.1053
j = 31	0.1358	0.1150	0.1782
j = 32	0.3910	0.1540	0.1782
j = 33	0.2816	0.0500	0.0
j = 34	0.1974	0.0490	0.3483
j = 35	1.0607	0.0220	0.0
j = 36	0.3951	0.2390	0.0
j = 37	0.0	0.0	0.0
j = 38	0.0	0 0	0.0
j = 39	0.0	0.0	0.0
j = 40	1.0000	0.0	0.0
j = 41	0.5639	0.0	0.0
j = 42	0.0	0.0	0.0
j = 43	0.0	0.0	0.0
j = 44	0.2330	0.0460	0.0
J = 45	0.4726	0.1530	0.1782

P_{ij}	i = 1	i = 2	i = 3	i = 4
j = 46	0.3137	0.1527	0.1856	0.3533
j = 47	0.0876	0.2337	0.2337	0.2337
j = 48	0.0819	0.0593	0.0471	0.0932
j = 49	0.0	0.0	0.1641	0.1641
j = 50	0.8139	0.1934	0.1934	0.1934
j = 51	0.2565	0.4606	0.1504	0.2945
j = 52	0.8162	0.4058	0.4058	0.4058
j = 53	0.2569	0.6677	0.1804	0.3529
j = 54	1.3428	0.5631	0.7030	0.7030
j = 55	1.3680	0.5612	0.5612	1.2562
j = 56	0.0	0.0	0.0	0.0
j = 57	0.4781	0.5573	0.5573	0.5573
j = 58	0.0805	0.0277	0.0457	0.0919
j = 59	0.8097	0.0637	0.0637	0.0637
j = 60	0.2510	0.2293	0.1421	0.2830
j = 61	0.0772	0.1120	0.1120	0.1120

P_{ij}	i = 5	i = 6	i = 7
j = 46	0.2707	0.0	0.2683
j = 47	0.2337	0.0	0.2421
j = 48	0.0819	0.0	0.0823
j = 49	0.5366	0.0	0.2176
j = 50	0.8148	0.0	0.8171
j = 51	0.4076	0.0	0.4276
j = 52	0.8176	0.0	0.8207
j = 53	0.6316	0.0	0.6551
j = 54	1.2897	0.0	0.7351
j = 55	1.2234	0.0	0.7924
j = 56	0.0	0.7353	0.0
j = 57	0.5573	0.0	0.6876
j = 58	0.0798	0.0	0.0799
j = 59	0.0895	0.0	0.8100
j = 60	0.2466	0.0	0.2469
j = 61	0.1120	0.0	0.1162

P_{ij}	i = 8	i = 9	i = 10
j = 46	0.1324	0.3446	0.2868
j = 47	0.1263	0.1263	0.1322
j = 48	0.0284	0.0920	0.0844
j = 49	0.1443	0.1443	0.4848
j = 50	0.1803	0.1803	0.8204
j = 51	0.3479	0.2922	0.2678
j = 52	0.3806	0.3806	0.8244
j = 53	0.5443	0.3496	0.4842
j = 54	0.6039	1.4349	0.7827
j = 55	0.5474	1.3944	1.0718
j = 56	0.0	0.0	0.0
j = 57	0.5145	0.5145	0.5287
j = 58	0.0132	0.0911	0.0815
j = 59	0.0565	0.0565	0.8122
j = 60	0.1643	0.2804	0.2553
j = 61	0.0608	0.0697	0.0646

P_{ij}	i = 11	i = 12	i = 13
j = 46	0.2465	0.1620	0.1782
j = 47	0.2641	0.0640	0.0745
j = 48	0.0842	0.0344	0.0745
j = 49	0.4547	0.1790	0.0
j = 50	0.8714	0.2243	0.7630
j = 51	0.5516	0.1415	0.2308
J = 52	0.8387	0.1518	0.7630
j = 53	0.7057	0.2870	0.2308
j = 54	0.8392	0.1900	0.0
j = 55	0.6240	0.1460	0.0
j = 56	0.0	0.0	0.0
j = 57	0.7425	0.2220	0.3896
j = 58	0.0827	0.0178	0.0340
j = 59	0.8134	0.0245	0.3483
j = 60	0.2914	0.0772	0.1053
j = 61	0.1245	0.0348	0.0340

Problem Data

	T_j	D_j	l_j	C_j	V_j
j = 1	500.0000	0.0	0.0	0.1000	20.0000
j = 2	500.0000	0.0	0.0	0.2500	1.0000
j = 3	500.0000	0.0	0.0	0.1500	1.0000
j = 4	500.0000	0.0	0.0	0.1500	1.0000
j = 5	500.0000	0.0	0.0	0.2000	1.0000
j = 6	500.0000	0.0	0.0	0.2000	1.0000
j = 7	500.0000	0.0	0.0	0.1000	1.0000
j = 8	500.0000	0.0	0.0	0.1500	1.0000
j = 9	500.0000	0.0	0.0	0.1000	10.0000
j = 10	300.0000	0.0	0.0	0.1000	10.0000
j = 11	300.0000	0.0	0.0	0.1000	10.0000
j = 12	300.0000	0.0	0.0	0.1000	1.0000
j = 13	300.0000	0.0	0.0	0.1000	1.0000
j = 14	300.0000	0.0	0.0	0.5000	15.0000
j = 15	300.0000	0.0	0.0	0.5000	1.0000
j = 16	300.0000	0.0	0.0	0.1000	10.0000
j = 17	300.0000	0.0	0.0	0.2000	10.0000
j = 18	500.0000	0.0	0.0	0.2000	10.0000
j = 19	500.0000	0.0	0.0	0.5000	20.0000
j = 20	500.0000	0.0	0.0	0.4500	20.0000
j = 21	500.0000	0.0	0.0	0.5000	12.0000
j = 22	500.0000	0.0	0.0	0.1500	12.0000
j = 23	500.0000	0.0	0.0	0.5500	20.0000
j = 24	900.0000	0.0	0.0	0.5000	10.0000
j = 25	900.0000	0.0	0.0	0.5000	10.0000
j = 26	900.0000	0.0	0.0	0.5000	9.0000
j = 27	900.0000	0.0	0.0	0.1000	10.0000
j = 28	900.0000	0.0	0.0	0.1000	6.0000
j = 29	900.0000	0.0	0.0	0.3000	10.0000
j = 30	900.0000	0.0	0.0	0.2500	10.0000
j = 31	900.0000	0.0	0.0	0.5000	1.0000
j = 32	1500.0000	0.0	0.0	0.3000	1.0000
j = 33	1500.0000	0.0	0.0	0.4000	1.0000
j = 34	500.0000	0.0	0.0	0.4000	15.0000
j = 35	500.0000	0.0	0.0	0.4000	1.0000
j = 36	500.0000	0.0	0.0	0.4000	1.0000
j = 37	500.0000	0.0	0.0	0.4000	15.0000
j = 38	500.0000	0.0	0.0	0.4000	15.0000
j = 39	700.0000	0.0	0.0	0.4000	12.0000
j = 40	700.0000	0.0	0.0	0.2500	5.0000
j = 41	700.0000	0.0	0.0	0.2500	5.0000
j = 42	700.0000	0.0	0.0	0.4000	15.0000
j = 43	700.0000	0.0	0.0	0.4000	15.0000
j = 44	700.0000	0.0	0.0	0.2000	10.0000
j = 45	400.0000	0.0	0.0	0.2000	10.0000

	T_j	D_j	l_j	C_j	V_j
j = 46	400.0000	0.0	0.0	0.2000	10.0000
j = 47	400.0000	0.0	0.0	0.2000	2.0000
j = 48	400.0000	0.0	0.0	0.2000	10.0000
j = 49	400.0000	0.0	0.0	0.4000	10.0000
j = 50	400.0000	0.0	0.0	0.2500	20.0000
j = 51	400.0000	0.0	0.0	0.2500	10.0000
j = 52	5000.0000	0.0	0.0	0.3500	2.0000
j = 53	500.0000	0.0	0.0	0.3000	2.0000
j = 54	500.0000	0.0	0.0	0.4500	10.0000
j = 55	500.0000	0.0	0.0	0.4500	10.0000
j = 56	500.0000	0.0	0.0	0.4500	10.0000
j = 57	900.0000	0.0	0.0	0.2500	10.0000
j = 58	900.0000	0.0	0.0	0.4000	10.0000
j = 59	900.0000	0.0	0.0	0.2000	2.0000
j = 60	300.0000	0.0	0.0	0.2000	2.0000
j = 61	300.0000	0.0	0.0	0.4000	2.0000

Results for Problem 10: (XS)

(only non-zero values of final solution shown)

f(x)	-0.20087D+06		
$x_{1,29}$	720.33	$x_{3,54}$	570.49
$x_{1,31}$	95.000	$x_{3,55}$	685.66
$x_{1,44}$	2388.6	$x_{3,57}$	21.444
$x_{1,46}$	819.53	$x_{4,1}$	487.50
$x_{1,50}$	11.342	$x_{4,18}$	1708.3
$x_{1,52}$	7.1002	$x_{4,27}$	1414.3
$x_{1,54}$	33.031	$x_{4,30}$	1009.1
$x_{1,55}$	159.98	$x_{4,46}$	130.84
$x_{1,58}$	106.53	$x_{5,17}$	163.12
$x_{1,59}$	408.59	$x_{5,20}$	857.98
$x_{2,19}$	359.81	$x_{5,22}$	122.16
$x_{2,40}$	538.20	$x_{5,29}$	125.73
$x_{2,45}$	1215.4	$x_{5,48}$	113.00
$x_{2,51}$	449.35	$x_{5,49}$	574.18
$x_{2,53}$	263.15	$x_{5,50}$	186.12
$x_{2,55}$	87.235	$x_{5,52}$	1814.3
$x_{2,57}$	1836.9	$x_{5,53}$	43.287
$x_{3,1}$	75.840	$x_{5,54}$	169.69
$x_{3,10}$	514.79	$x_{5,55}$	9.5330
$x_{3,14}$	1099.7	$x_{5,58}$	456.53
$x_{3,19}$	111.06	$x_{5,59}$	114.33
$x_{3,22}$	252.46	$x_{6,38}$	935.99
$x_{3,27}$	598.34	$x_{6,39}$	962.27
$x_{3,35}$	466.32	$x_{6,42}$	1084.8
$x_{3,40}$	290.90	$x_{6,43}$	1187.7
$x_{3,45}$	63.036	$x_{6,56}$	579.31

$x_{7,20}$	268.36	$x_{11,27}$	149.05
$x_{7,22}$	129.81	$x_{11,28}$	3666.1
$x_{7,29}$	436.42	$x_{11,51}$	111.91
$x_{7,32}$	95.000	$x_{12,17}$	2491.4
$x_{7,41}$	111.42	$x_{12,19}$	932.12
$x_{7,44}$	245.28	$x_{12,20}$	607.74
$x_{7,48}$	804.67	$x_{13,23}$	95.000
$x_{7,51}$	36.377	$x_{13,46}$	623.77
$x_{7,52}$	2316.2	$x_{13,34}$	1833.3
$x_{7,53}$	59.708	$x_{13,48}$	112.45
$x_{7,59}$	246.82	$x_{13,50}$	392.41
$x_{8,9}$	1197.7	$x_{13,52}$	2411.9
$x_{8,28}$	748.63	g_1	4750.0
$x_{8,40}$	56.122	g_2	4750.0
$x_{8,41}$	1594.3	g_3	4750.0
$x_{8,51}$	506.04	g_4	4750.0
$x_{8,53}$	647.25	g_5	4750.0
$x_{9,1}$	278.91	g_6	4750.0
$x_{9,11}$	5.3503	g_7	4750.0
$x_{9,16}$	1446.6	g_8	4750.0
$x_{9,27}$	757.37	g_9	4750.0
$x_{9,30}$	2140.1	g_{10}	4750.0
$x_{9,46}$	121.65	g_{11}	4750.0
$x_{10,20}$	658.93	g_{12}	4750.0
$x_{10,29}$	41.692	g_{13}	4750.0
$x_{10,48}$	1120.5	g_{14}	526.80
$x_{10,49}$	418.16	g_{22}	144.25
$x_{10,52}$	1120.5	g_{23}	316.08
$x_{10,58}$	921.35	g_{24}	0.48313
$x_{10,59}$	468.90	g_{27}	415.893
$x_{11,9}$	727.91	g_{29}	316.08
$x_{11,24}$	31.299	g_{30}	334.72
$x_{11,25}$	63.701	g_{31}	170.31

g_{32}	693.15
g_{33}	664.26
g_{35}	541.73
g_{36}	0.38000
g_{37}	0.53835
g_{38}	1.0957
g_{40}	948.24
g_{41}	831.35
g_{42}	1070.0
g_{43}	1035.7
g_{44}	11.257
g_{45}	24.614
g_{47}	638.53
g_{48}	638.53
g_{51}	638.53
g_{52}	644.92
g_{53}	805.51
g_{54}	805.51
g_{55}	616.04
g_{56}	893.94
g_{57}	781.00
g_{58}	446.29
g_{59}	446.29
g_{61}	178.42
g_{62}	510.83
g_{63}	460.29
g_{64}	460.29
g_{65}	6154.0
g_{66}	594.46
g_{67}	664.26
g_{68}	664.26
g_{69}	425.97
g_{70}	1035.7
g_{71}	120.10
g_{72}	1004.1
g_{75}	712.50
g_{76}	1153.3
g_{77}	95.000
g_{78}	95.000
g_{79}	268.36
g_{80}	95.000
g_{81}	95.000

Problem 11

See Section III.C.11.

Problem 12

See Problem 6.

Problem 13 (Dean--Integer Nonlinear)

Source: [Ref. 38]

No. of independent variables: 12 (8 binary)

No. of constraints: 48 nonlinear equality constraint

3 linear equality constraint

Objective function:

$$\text{minimize:} \quad \sum_{i=1}^{I} \rho_i(2\delta_i n_i)$$

Constraints:

$$\frac{\varepsilon_{1ik}}{\varepsilon_{L_i}^{T}} \leq 1, \qquad i = 1,\ldots,I;\ k = 1,\ldots,k;$$

$$\frac{\varepsilon_{1ik}}{\varepsilon_{L_i}^{C}} \leq 1, \qquad i = 1,\ldots,I;\ k = 1,\ldots,k;$$

$$\frac{\varepsilon_{2ik}}{\varepsilon_{T_i}^{T}} \leq 1, \qquad i = 1,\ldots,I;\ k = 1,\ldots,k;$$

$$\frac{\varepsilon_{2ik}}{\varepsilon_{T_i}^{C}} \leq 1, \qquad i = 1,\ldots,I;\ k = 1,\ldots,k;$$

$$\frac{\gamma_{12ik}}{\gamma_{LT}^{+}} \leq 1, \qquad i = 1,\ldots,I;\ k = 1,\ldots,k;$$

$$\frac{\gamma_{12ik}}{\gamma_{LT}^{-}} \leq 1, \qquad i = 1,\ldots,I;\ k = 1,\ldots,k;$$

$$\sum_{i=1}^{I} (c'_{11})_i \, (2\delta_i n_i) \geq A_{11}^{(1)}$$

$$\sum_{i=1}^{I} (c'_{22})_i \, (2\delta_i n_i) \geq A_{22}^{(1)}$$

$$\sum_{i=1}^{I} (c'_{66})_i \, (2\delta_i n_i) \geq A_{66}^{(1)}$$

$$n_i \geq 0, \qquad i = 1....,I$$

$$0 \leq \theta_i \leq 90°, \qquad i = 1,...,I$$

where:

$$\theta_i = x_i, \qquad i = 1,...,I$$

$$n_i = x_j + 2x_{j+1} + 4x_{j+2}, \quad (j = 3(i - 1) + I)$$

$$\varepsilon_{1ik} = l_i^2 \varepsilon_{xk} + m_i^2 \varepsilon_{yk} + m_i l_i \gamma_{xyk}$$

$$\varepsilon_{2ik} = m_i^2 \varepsilon_{xk} + l_i^2 \varepsilon_{xk} - m_i l_i \gamma_{xyk}$$

$$\gamma_{12ik} = -2m_i l_i \varepsilon_{xk} + 2m_i l_i \varepsilon_{yk} + (l_i^2 - m_i^2) \gamma_{xyk}$$

and where:

$$\varepsilon_{xk} = \frac{D_1}{D}$$

$$\varepsilon_{yk} = \frac{D_2}{D}$$

$$\gamma_{xyk} = \frac{D_3}{D}$$

and where:

$$D = A_{11}(A_{22}A_{66} - A_{26}^2) - A_{12}(A_{12}A_{66} - A_{26}A_{16}) + A_{16}(A_{12}A_{26} - A_{22}A_{16})$$

$$D_1 = N_{xk}(A_{22}A_{66} - A_{26}^2) - A_{12}(N_{yk}A_{66} - A_{26}N_{xyk}) + A_{16}(N_{yk}A_{26} - A_{22}N_{xyk})$$

$$D_2 = A_{11}(N_{yk}A_{66} - A_{26}N_{xyk}) - N_{xk}(A_{12}A_{66} - A_{26}A_{16}) + A_{16}(A_{12}N_{xyk} - N_{yk}A_{16})$$

$$D_3 = A_{11}(A_{22}N_{xyk} - N_{yk}A_{26}) - A_{12}(A_{12}N_{xyk} - N_{yk}A_{16}) + N_{xk}(A_{12}A_{26} - A_{22}A_{16})$$

and where:

$$A_{rs} = \sum_{i=1}^{I} (c'_{rs})_i (2\delta_i n_i), \qquad r,s = 1,2,6$$

and where:

$$(C'_{11})_i = (C_{11})_i l_i^4 + 2(C_{12})_i l_i^2 m_i^2 + (C_{22})_i m_i^4 + 4(C_{66})_i m_i^2 l_i^2$$

$$(C'_{12})_i = (C_{11})_i l_i^2 m_i^2 + (C_{12})_i (l_i^4 + m_i^4) + (C_{22})_i l_i^2 m_i^2 - 4(C_{66})_i l_i^2 m_i^2$$

$$(C'_{16})_i = (C_{11})_i l_i^3 m_i + (C_{12})_i (m_i^3 l_i - l_i^3 m_i) - (C_{22})_i m_i^3 l_i + 2(C_{66})_i (m_i^3 l_i - m_i l_i^3)$$

$$(C'_{22})_i = (C_{11})_i m_i^4 + 2(C_{12})_i m_i^2 l_i^2 + (C_{22})_i l_i^4 + 4(C_{66})_i m_i^2 l_i^2$$

$$(C'_{26})_i = (C_{11})_i m_i^3 l_i + (C_{12})_i (l_i^3 m_i - m_i^3 l_i - (C_{22})_i m_i l_i^3 + 2(C_{66})_i (m_i l_i^3 - m_i^3 l_i)$$

$$(C'_{66})_i = (C_{11})_i m_i^2 l_i^2 - 2(C_{12})_i m_i^2 l_i^2 + (C_{22})_i m_i^2 l_i^2 + (C_{66})_i (l_i^2 - m_i^2)^2$$

and where:

$$(C_{11})_i = \frac{E_{Li}}{(1 - \nu_{LTi}\nu_{TLi})}$$

$$(C_{12})_i = \frac{\nu_{TLi}E_{Li}}{(1 - \nu_{LTi}\nu_{TLi})} = \frac{\nu_{LTi}\nu_{Ti}}{(1 - \nu_{LTi}\nu_{TLi})}$$

$$(C_{22})_i = \frac{E_{Ti}}{(1 - \nu_{LTi}\nu_{TLi})}$$

$$(C_{66})_i = G_{LTi}$$

and where:

$$\frac{\nu_{LTi}}{E_{Li}} = \frac{LT\nu_i}{E_{Ti}}$$

Problem data:

I = 4; K = 3

$A_{11}^{(1)}$ = 0.75D06

$A_{22}^{(1)}$ = 0.50D06

$A_{66}^{(1)}$ = 0.50D06

δ_i = 0.005, i = 1,...,4

ρ_i = 1, i = 1,...,4

E_{Li} = 20.0D06, i = 1,...,4

E_{Ti} = 1.3D06, i = 1,...,4

G_{LTi} = 0.65D06, i = 1,...,4

ν_{LTi} = 0.304, i = 1,...,4

ε_{Li}^{T} = 8.25D-03, i = 1,...,4

ε_{L}^{C} = -5.75D-03, i = 1,...,4

ε_{Ti}^{T} = 6.15D-03, i = 1,...,4

ε_{Li}^{C} = 2.306D-02 i = 1,...,4

$\gamma_{LTi}^{+} = \gamma_{LTi}^{-}$ = 2.46D-02 i = 1,...,4

N_{x1} = 4000.0

N_{y1} = 1000.0

N_{xy1} = 2000.0

N_{x2} = -3000.0

N_{y2} = 1000.0

N_{xy2} = -2000.0

N_{x3} = 2000.0

N_{y3} = 1000.0

N_{xy3} = 3000.0

Results for Problem 13:
(only final solution shown)

f(x)	0.12000		
n_1	5.4978	g_{21}	0.45455
n_2	5.8905	g_{22}	-0.12123
n_3	6.7285	g_{23}	-0.26403
n_4	7.1540	g_{24}	-0.26403
θ_{11}	6.0000	g_{25}	0.23177
θ_{12}	0.0	g_{26}	-0.33254
θ_{21}	6.0000	g_{27}	-0.19951
θ_{22}	0.0	g_{28}	0.53207D-01
θ_{31}	6.0000	g_{29}	-0.45748
θ_{32}	0.0	g_{30}	-0.45748
θ_{41}	6.0000	g_{31}	-0.24848
θ_{42}	0.0	g_{32}	0.35651
g_1	-0.70122D-02	g_{33}	-0.63711
g_2	0.10061D-01	g_{34}	0.16992
g_3	0.20118	g_{35}	-0.45286
g_4	-0.53655D-01	g_{36}	-0.45286
g_5	0.23862	g_{37}	-0.61069
g_6	0.23862	g_{38}	0.87621
g_7	0.25052	g_{39}	-0.71607
g_8	-0.35944	g_{40}	0.19097
g_9	0.55526	g_{41}	0.24063
g_{10}	-0.14809	g_{42}	0.24063
g_{11}	0.28885	g_{43}	-0.20712
g_{12}	0.28805	g_{44}	0.29717
g_{13}	0.60751	g_{45}	-0.58747
g_{14}	-0.87164	g_{46}	0.15667
g_{15}	0.56892	g_{47}	0.48192
g_{16}	-0.15173	g_{48}	0.48192
g_{17}	-0.78705D-01	g_{49}	-0.20411
g_{18}	-0.78705D-01	g_{50}	0.29286
g_{19}	0.44638	g_{51}	0.42399
g_{20}	-0.64045	g_{52}	-0.11308

g_{53}	0.67886D-01
g_{54}	0.67886D-01
g_{55}	-0.56198D-01
g_{56}	0.80633D-01
g_{57}	0.47633
g_{58}	-0.12703
g_{59}	0.21884
g_{60}	0.21884
g_{61}	0.47443
g_{62}	-0.68071
g_{63}	-0.19633D-01
g_{64}	0.52360D-02
g_{65}	0.99156D-01
g_{66}	0.99156D-01
g_{67}	0.49857
g_{68}	-0.71534
g_{69}	-0.13880
g_{70}	0.37017D-01
g_{71}	-0.10797
g_{72}	-0.10797
g_{73}	1.1737
g_{74}	0.51938
g_{75}	0.50936

LIST OF REFERENCES

1. Pierskalla, W., and Ratliff, H. D., "Reporting Computational Experiences in Operations Research," Operations Research, v. 29, pp. xi-xiv, March-April 1981.

2. International Business Machines Report SC28-6852-2, IBM OS FORTRAN IV (H Extended) Compiler Programmer's Guide, November 1974.

3. International Business Machines Report SH20-0968-1, Mathematical Programming System-Extended (MPSX), and Generalized Upper Bounding, IBM Corporation, New York, 1974.

4. Systems Optimization Laboratory, Department of Operations Research, Stanford University Technical Report SOL 80-100, MINOS Distribution Documentation, by M. A. Saunders, September 1980.

5. Systems Optimization Laboratory, Department of Operations Research, Stanford University Technical Report SOL 77-9, MINOS User's Guide, by B. A. Murtagh and M. A. Saunders, February 1977.

6. Systems Optimization Laboratory, Department of Operations Research, Stanford University Technical Report SOL 80-14, MINOS/Augmented User's Manual, by B. A. Murtagh and M. A. Saunders, June 1980.

7. Dantzig, G. B., Linear Programming and Extensions, Princeton University Press, 1963.

8. Saunders, M. A., "A Fast, Stable Implementation of the Simplex Method Using Bartels-Golub Updating," in: Sparse Matrix Computations, edited by J. R. Bunch and D. J. Rose, Academic Press, 1976.

9. Hellerman, E. and Rarick, D. C., "The Partitioned Preassigned Pivot Procedure," in: Sparse Matrices and their Applications, edited by D. J. Rose and R. A. Willoughby, Plenum Press, 1972.

10. Bartels, R. H. and Golub, G. H., "The Simplex Method of Linear Programming Using LU Decomposition, Communications of the ACM, May 1969.

11. Wolfe, P., "Methods of Nonlinear Programming," in: Nonlinear Programming, edited by J. Abadie, North-Holland, 1967.

12. Fletcher, R. and Reeves, C. M., "Function Minimization by Conjugate Gradients," Computer Journal, July 1964.

13. Systems Optimization Laboratory, Department of Operations Research, Stanford University Technical Report SOL 80-1, The Implementation of a Lagrangian-based Algorithm for Sparse Nonlinear Constraints, by B. A. Murtagh and M. A. Saunders, May 1980.

14. Brown, G. G. and Graves, G. W., "Elastic Programming: A New Approach to Large-Scale Mixed-Integer Optimization," Mixed Integer Programming, paper presented at the ORSA/TIMS meeting, Las Vegas, Nevada, 17 November, 1975.

15. Brown, G. G., "Issues in Basic Manipulation: Factorization/Decomposition," The New Generation of L.P. Codes, paper presented at the ORSA/TIMS meeting, Miami, Florida, 5 November, 1976.

16. Graves, G. W., "A Complete Constructive Algorithm for the General Mixed Linear Programming Problem," Naval Research Logistics Quarterly, March, 1965.

17. Graves, G. W., and McBride, R. D.,, "The Factorization Approach to Large-Scale Linear Programming," Mathematical Programming, February 1976.

18. American National Standard Institute (ANSI), Standard Publication X3.9, (commonly known as FORTRAN 77), 1978.

19. Graves, G. W. and Whinston, A. B., "The Application of a Nonlinear Programming Algorithm to a Second Order Representation of the Problem," Cahiers du Centre D'Etudes de Recherche Operationnelle, v. 11, n. 2, 1969.

20. Brown, G. and Graves, G., XS Mathematical Programming System, perpetual working paper, (c. May 1980).

21. Graves. G. W., Mathematical Programming, unpublished monograph (available only from the author), (c. 1981).

22. Insight, Inc., ODS-Optimization for Distribution Systems: Systems Manual, 1978.

23. Insight, Inc., ODS-Optimization for Distribution Systems: User's Guide, 1978.

24. Naval Postgraduate School Technical Report NPS52-80-006, ATHENA: User's Manual for Interactive Analysis of Large-Scale Optimization Models, by G. H. Bradley, G. G. Brown, and P. I. Galatas, April 1980.

25. Waterman, R. J., An Evaluation and Comparison of Three Nonlinear Programming Codes, Master's Thesis, Naval Postgraduate School, March 1976.

26. Himmelblau, D. M., Applied Nonlinear Programming, McGraw-Hill, 1972.

27. Choe, U. C. and Schrady, D. A., "Models for Multi-item Continuous Review Inventory Policies Subject to Constraints," Naval Research Logistics Quarterly, v. 18, n. 4, December 1971.

28. Scott, A. J., "A Model of Nodal Entropy in a Transportation Network with Congestion Costs," Transportation Science, May 1971.

29. Manne, A. S., ETA-MACRO: A Model of Energy-Economy Interactions, in Modeling Energy-Economy Interactions, edited by C. J. Hitch, Resources for the Future, Washington, D.C., 1977.

30. Research Analysis Corporation Report RAC-TP-272, The Chemical Equilibrium Problem: An Application of SUMT, by A. P. Jones, 1967.

31. Dantzig, G. B., Johnson, S. M., and White, W. B., "Chemical Equilibrium in Complex Mixtures," Journal of Chemical Physics, May 1958.

32. Bracken, J. and McCormick, G. P., Selected Applications of Nonlinear Programming, John Wiley & Sons, Inc., 1968.

33. International Business Machines New York Technical Center Technical Report 320-2949, A Comparative Study on Nonlinear Programming Codes, by A. R. Colville, June 1968.

34. Research Analysis Corporation Report RAC-TP-302, On Variable Metric Methods of Minimization, by J. D. Pearson, May 1968.

35. Paviani, D. A., A New Method for the Solution of the General Nonlinear Programming Problem, Ph.D. dissertation, University of Texas, Austin, 1969.

36. Mylander, W. C. III, "Applied Mathematical Programming," Proceedings of the U.S. Army Operations Research Symposium, Part I, March 1965.

37. Rand Corporation Report R-1411-DDPAE, Sortie Allocation by a Nonlinear Programming Model for Determining a Munitions Mix, by R. J. Clasen, G. W. Graves, and J. Y. Lu, March 1974.

38. Farshi, B. and Schmit, L. A., "Optimum Laminate Design for Strength and Stiffness," International Journal for Numerical Methods in Engineering, v. 7, N. 4, 1973.

39. Brown, G. G., Dean, D. R., and Graves, G. W., "Computational Experiments with Large-Scale Nonlinear Optimization," Large-Scale Nonlinear Programming, presented at CORS/ORSA/TIMS meeting, Toronto, Canada, 5 May 1981.

INITIAL DISTRIBUTION LIST

1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
3.	Department Chairman, Code 55 Department of Operations Research Naval Postgraduate School Monterey, California 93940	1
4.	Professor Gerald G. Brown, Code 55bw Department of Operations Research Naval Postgraduate School Monterey, California 93940	65
5.	LCDR Dennis R. Dean, USN Pre-Commissioning Unit USS Ticonderoga (CG-47) FPO New York, New York 09501	1

Thesis
D1827 Dean
c.1
194421
Computational advances in large-scale nonlinear optimization.